BIBLIOTECA BASICA INFORMATICA

BANCOS DE DATOS (III)

22



INGELEK

BIBLIOTECA BASICA INFORMATICA

BANCOS DE DATOS (II)

22

INGELEK

Director editor:
Antonio M. Ferrer Abelló.

Director de producción:
Vicente Robles.

Coordinador y supervisión técnica:
Enrique Monsalve.

Colaboradores:
Angel Segado
Casimiro Zaragoza
Fernando Ruíz
Francisco Ruíz
Jesús Pedraza
Juanjo Alba Ríos
Margarita Caffaratto
María Angeles Gálvez
Marina Caffaratto
Masé González Balandín
Patricia Mordini

Diseño:
Bravo/Lofish.

Dibujos:
José Ochoa.

ISBN del tomo: 84-85831-55-1
ISBN de la obra: 84-85831-31-4
Fotocomposición: Pérez Díaz, S. A.
Imprime: Héroes, S. A.
Depósito Legal: M-4862-1986
Precio en Canarias, Ceuta y Melilla: 380 pts.



INDICE

PROLOGO



Por razones de espacio nos hemos visto obligados a dividir el tema de los bancos y bases de datos en BASIC y Pascal en dos volúmenes. En este segundo y último podrán también seguir todos los programas en los apéndices.

Al igual que en el anterior volumen iremos comentando cada uno de los programas y subrutinas necesarios para obtener, tanto en BASIC como en Pascal, la gestión de datos mediante su introducción, actualización y acceso permanente.

Quizá considere al seguir las explicaciones que somos demasiado repetitivos en algunos casos, pero hemos preferido que en todo lugar quedara perfectamente claro lo que el programa pretendía lograr y cómo lo conseguía. Por lo demás, la estructura es la misma que la que pudo observar en el primer volumen.

Y ya sin más preámbulos pasemos a ver todo aquello que nos quedó pendiente.

CAPITULO I

ACCESO MEDIANTE CLAVE

Este tipo de organización nos permite tener acceso a cualquier registro, tanto por medio de una clave como accediendo al enésimo registro. La búsqueda mediante clave puede ser también parcial. Las claves están almacenadas en un fichero organizado en árbol binario.

Comentaremos tres programas: el primero, IS_PREPARA, define los dos ficheros; el segundo programa, IS_GESTION, nos da la posibilidad de insertar nuevos registros, modificarlos, borrarlos o leerlos y el último programa, IS_COMPACTA, vuelve a conformar los ficheros eliminando los registros borrados.

La lista de campos de los registros del fichero CLIENTES es la siguiente:

CLIENTE	campo alfabético de 31 caracteres
DIRECCION	campo alfabético de 33 caracteres
CIUDAD	campo alfabético de 15 caracteres
TELEFONO	campo numérico de 8 caracteres
PREFIJO	campo alfabético de 4 caracteres
TIPO	campo alfabético de 1 carácter

El primer registro del fichero CLIENTES contiene dos informaciones: la primera indica el número del registro que debe escribirse y la segunda indica cuántos registros se pueden grabar en el fichero.

La lista de campos del fichero INDICE es la siguiente:

NOMBRE	campo alfabético de 31 caracteres
IZQUIERDO	campo numérico de 2 caracteres
DERECHO	campo numérico de 2 caracteres

El campo izquierdo, si contiene un valor diferente de 0, indica que la clave de ese registro va precedida por otra clave. El campo derecho, si contiene un valor negativo, significa que la clave de ese registro va seguida por otra clave que se encuentra en el registro cuyo número corresponde al valor absoluto del campo derecho, cuyo registro se recorrió ya en parte; si contiene un valor positivo, significa que la clave de dicho registro irá seguida por la clave que se encuentra en ese número de registro, y, finalmente, si contiene el valor 0 (NIL) significa que no existen otros registros.

Programa IS_PREPARA

El programa pregunta cuántos registros serán escritos en el fichero como máximo. No siempre sabremos con exactitud dicho valor, pero podemos realizar una estimación y aumentarla en un 20 por 100 por precaución. Si en la utilización nos percatamos de haber fallado en las previsiones, podemos generar un fichero más amplio siguiendo las instrucciones de la explicación del programa IS_COMPACTA. Dicho programa tiene como objeto principal recuperar el espacio libre dejado al borrar registros.

El programa escribe en el primer registro dos informaciones. En el primer campo escribimos 2; de hecho, la primera información se escribirá en el registro 2. En el segundo campo llevamos el número de registros que puede contener el fichero. Luego escribimos todos los registros posibles, con los campos alfabéticos vacíos (a excepción del prefijo, que se coloca en 0000) y los campos numéricos a 0. A continuación abrimos el fichero INDICE y el campo del nombre se pone a un valor nulo, mientras que los campos izquierdo y derecho se ponen a 0, o sea con "NIL".

Entonces grabamos un número de registros igual al del fichero CLIENTES. El programa en BASIC IS_PREPARA está incluido en el apéndice reservado al BASIC. Veamos las instrucciones principales:

```
140 FILE$="CLIENTES.DAT"
160 OPEN "R",1,FILE$,88
```


Abrimos el fichero CLIENTES.DAT declarándolo de tipo aleatorio. El registro contendrá, como máximo, 88 caracteres.

```
170 INPUT "¿DE CUANTOS REGISTROS ESTARA
    COMPUESTO EL FICHERO?";ULTIMO
180 PRIMERO=2
190 FIELD 1,2 AS PRIMERO$,2 AS ULTIMO$
```

Después de haber pedido al usuario cuántos podrán ser los registros del fichero indicamos las características del registro mediante un FIELD; en este caso, estará constituido por dos campos con dos caracteres cada uno de ellos. Este es el primer registro, que nos indica el número del siguiente registro a grabar y cuántos registros podrá contener el fichero como máximo.

```
200 LSET PRIMERO$=MKI$(PRIMERO)
210 LSET ULTIMO$=MKI$(ULTIMO)
```

Mediante estas instrucciones transferimos al buffer (zona de memoria reservada para admitir los datos del fichero) los contenidos de las variables PRIMERO y ULTIMO. Son necesarias las instrucciones MKI$ para convertir un dato numérico en una cadena. A riesgo de ser pesados, repetimos que las instrucciones LSET son totalmente necesarias; si, por ejemplo, escribimos PRIMERO$=MKI$(PRIMERO) la instrucción será admitida por el BASIC, pero los datos no serán llevados al registro ni tampoco al fichero. ¡No lo olvidemos!

```
220 PUT #1,1
```

Escribimos a continuación el primer registro del fichero en la forma siguiente:

```
230 FIELD 1,31 AS CLIENTE$,33 AS DIRECCION$,
    15 AS CIUDAD$, 4 AS TELEFONO$,4 AS PREFIJO$,
    1 AS TIPO$
```

Declaramos las características de los sucesivos campos del fichero CLIENTES:

```
250 LSET CLIENTE$=""
```

Dejamos vacío el espacio destinado a contener el campo del cliente; a continuación siguen otras instrucciones para inicializar adecuadamente los demás campos.

```
320 FOR X%=2 TO ULTIMO+1
330 PUT #1,X%
340 NEXT X%
```

Activa un bucle de instrucciones que graba los registros declarados por el usuario, estando adecuadamente inicializado cada campo del registro. Después de haber cerrado el fichero CLIENTES se abre el fichero INDICE.DAT., en donde se grabarán los registros con los campos oportunamente inicializados.

```
400 OPEN "R",2,"INDICE.DAT",35
410 FIELD 2,31 AS NOMBRE$,2 AS IZQUIERDO$,2 AS DERECHO$
```

Se abre un fichero de tipo aleatorio constituido por tres campos.

```
420 LSET NOMBRE$=""
430 LSET IZQUIERDO$=MKI$(0)
440 LSET DERECHO$=MKI$(0)
```

Los tres campos del registro se inicializan de forma adecuada.

```
450 FOR X%=I TO ULTIMO
460 PUT #2,X%
470 NEXT X%
```

En el fichero se graba un número de registro igual al valor contenido en la variable ULTIMO, que es, en la práctica, el indicado por el usuario.

El programa Pascal denominado IS_PREPARA está incluido en el apéndice reservado al lenguaje Pascal; vamos a comentar también las instrucciones principales.

```
REWRITE(cliente,'cliente.dat');
WRITE('¿de cuantos ficheros estara compuesto el fichero?:');
READLN(ultimo);
```

Después de haber abierto el fichero se pide al usuario el número de registro del fichero para almacenarlo en la variable ULTIMO. El Pascal tiene la posibilidad de declarar varios tipos de registros en el mismo fichero utilizando la palabra clave CASE en la declaración del registro correspondiente; obsérvese, a tal propósito, la declaración del tipo "CLI" en el programa IS_PREPARA antes citado.


```
primero:=2;
maximo:=ultimo+1;
PUT(cliente);
PUT(cliente);
```

Los dos campos del registro se inicializan, respectivamente, con 2 y con el valor proporcionado por el usuario; las instrucciones PUT graban los registros en disco.

```
cliente:='';
direccion:='';
ciudad:='';
telefono:=0;
prefijo:='0000';
tipo:=' ';
```

Todos los campos están puestos a 0 o vacíos.

```
FOR X:=1 TO ultimo DO PUT(cliente);
```

Se graba un número de registros igual a los declarados por el usuario. Después del cierre del fichero irán las instrucciones siguientes:

```
REWRITE
WITH indice^
DO BEGIN
     nombre:='';
     izquierdo:=0;
     derecho:=0;
     FOR x:=0 TO ultimo+1 DO PUT(indice);
END;
CLOSE(indice.LOCK);
```

que abren el fichero INDICE, el cual contendrá un número de registros igual al contenido por la variable ULTIMO; cada registro está constituido por tres campos oportunamente inicializados.

Programa IS_GESTION

El programa se realizó según las normas de la programación estructurada, subdividiéndolo en numerosas subrutinas. El núcleo fundamental del programa está reducido al mínimo. Un menú su-

ficientemente explicativo permite la elección entre varias opciones, cada una de las cuales hace uso de varias subrutinas. El programa principal se encarga de abrir el fichero INDICE y el fichero CLIENTES y de almacenar, en la variable ACTUAL, el número del primer registro a grabar, y en la variable ULTIMO, el número de los registros existentes en el fichero.

El programa está incluido en el apéndice reservado al BASIC con la denominación IS_GESTION; pasamos a comentar las instrucciones principales.

```
140 OPEN "R",1,"CLIENTES.DAT",88
```

Abrimos el fichero CLIENTES.DAT declarándolo de acceso aleatorio; cada uno de sus registros podrá contener hasta 88 caracteres.

```
180 FIELD 1,2 AS ACTUAL$,2 AS ULTIMO$
190 GOSUB 15000:REM  -->LECTURA CLIENTE/1
200 ACTUAL%=CVI(ACTUAL$)
```

Leemos el primer registro del fichero CLIENTES para almacenar en la variable ACTUAL% el número del primer registro a grabar y en la variable ULTIMO% el número de los registros que puede contener el fichero.

```
211 OPEN "R",2,"INDICE.DAT",25
```

Abrimos el fichero INDICE.DAT declarándolo de acceso aleatorio; cada uno de sus registros podrá contener hasta 35 caracteres.

```
230 GOSUB 800:REM ----->MENU
240 ON QUEQUIERE GOSUB 2000,4000,6000,9000
```

Cedemos el control a la subrutina MENU, que nos permite elegir una de las cinco funciones. Si se da la respuesta 1, el programa cederá el control a la subrutina que se inicia con la instrucción 2000; si se da la respuesta 2, el programa cederá el control a la subrutina que se inicia a partir de la instruccion 4000, y así sucesivamente.

```
250 IF QUEQUIERE<>9 THEN GOTO 220
270 CLOSE 1
280 CLOSE 2
290 END
```


Si a la solicitud del menú respondemos con el valor 9, se cerrarán los dos ficheros y acabará el programa; de no ser así, se visualizará el menú.

El programa IS_GESTION está incluido en el apéndice reservado al lenguaje Pascal; comentaremos las instrucciones principales.

```
RESET(indice,'indice.dat');
RESET(cliente,'cliente.dat');
```

Abrimos tanto el fichero INDICE como el fichero CLIENTES.

```
SEEK(cliente,1);
GET(cliente);
actual:=cliente^.primero;
ultimo:=cliente^.maximo;
```

Leemos el primer registro del fichero CLIENTES; ello nos permite almacenar en ACTUAL el número del primer registro a grabar, y en ULTIMO, el número de registros que puede contener el fichero.

```
REPEAT
  menu;
  CASE que_quiere OF
       '1':carga;
       '2':lectura;
       '3':para_nombre;
       '4':modificacion;
       '5':borrado;
     END;
```

Habrá comprendido ya que basándose en la respuesta dada se llamará una de las cinco funciones.

```
   UNTIL que_quiere='9';
   CLOSE (indice.LOCK);
   CLOSE(cliente.LOCK);
END.
```

Si a la solicitud del menú respondemos con el valor 9, se cierran los ficheros y se termina el programa.

En la descripción de las demás subrutinas seguiremos el orden indicado en el menú.

Subrutina CARGA

Después de borrar la pantalla se visualizará el número del primer registro que se grabará. Esto puede ser interesante cuando, al reanudar la carga, queramos saber cuántos registros hemos cargado ya y, por consiguiente, poder evaluar cuánto tiempo se empleará para cargar el resto.

Si estamos en la fase de carga avanzada, será útil saber cuál fue el último registro cargado y comprobar si la operación se realizó de forma correcta. A continuación, el control se cede a la subrutina CLAVE, que tiene como objeto buscar en el fichero INDICE el nombre del cliente.

En un fichero de acceso mediante clave es importante que no existan claves idénticas, porque si existieran nos sería imposible distinguirlas; por ello rechazaremos durante la carga las claves que existan ya. Al entrar en el bucle de instrucciones en donde se solicitan los datos del cliente, dicho bucle terminará cuando se responda con "zzzz" a la solicitud del nombre del cliente.

Si existe espacio en el disco se pedirán, a través de la subrutina TECLADO, los datos correspondientes al nombre, ciudad, número de teléfono y prefijo. Luego el registro se graba sobre el disco en el mismo lugar. Finalmente, el control se cede a la subrutina ACT_NUDO, que se encarga de escribir en el fichero INDICE. Se actualiza el primer registro del fichero CLIENTES para el almacenamiento del número del próximo registro a grabar.

Subrutina ACT_NUDO

El objeto de esta subrutina es insertar la clave en orden ascendente en el fichero CLIENTES. La primera información se escribe en el registro 1 y los punteros izquierdo y derecho se ponen a cero. Para las sucesivas informaciones se examina el árbol desde el comienzo y se compara el contenido de la variable NOMINATIVO con el contenido de la variable NOMBRE. Pueden verificarse dos casos: que el contenido de la variable NOMINATIVO sea mayor que el de la variable NOMBRE o bien que el de NOMINATIVO sea inferior al de NOMBRE.

En la primera circunstancia (NOMINATIVO>NOMBRE) pueden verificarse, a su vez, dos casos: si el puntero derecho contiene un valor mayor que 0, se continuará la búsqueda a partir del nudo indicado por éste. Si, por el contrario, el puntero derecho contiene un valor negativo o el valor 0, se almacenará NOMINATIVO en el nudo apuntado por "ACTUAL" y, además, se actualizará el nudo apuntado por "P".

En el segundo caso (NOMINATIVO<NOMBRE) pueden verificarse también los dos casos: si el puntero izquierdo contiene un valor diferente de 0, se seguirá la búsqueda a partir del nudo indicado por él y si contiene el valor 0, se almacenará NOMINATIVO en el nudo apuntado por la variable "ACTUAL", mientras que se actualizará el nudo apuntado por "P". Hay que destacar que el puntero izquierdo está puesto a cero mientras que el puntero derecho contenga un valor negativo. Veamos con más detalle esta subrutina.

Se introduce en el programa IS_GESTION, en donde ocupa las líneas 12000 a 12540. Vamos a comentar las instrucciones principales:

```
12040 HECHO=0
12050 IF ACTUAL%=2 THEN LSET SI$=MKI$(0):LSET DE$=
      MKI$(0):LSET NO$=NOMINATIVO$:PUT#2,2:GOTO 12520
```

Si se tiene que grabar el primer nudo, es decir, la raíz del árbol, se ponen a 0 los dos punteros, se escribe el registro y se hace una nueva entrada desde la subrutina. De no ser así, se tendrá:

```
12080 NUMERO%=P%
12090 GOSUB 16000:REM -->LECTURA INDICE
12100 IF NOMINATIVO$>NO$ THEN GOTO 12300
```

Después de haber posicionado de forma adecuada la variable P%, se da comienzo a un bucle de instrucciones para la búsqueda del nudo en donde ha de insertarse el nuevo NOMINATIVO. En caso de que el NOMINATIVO sea "mayor" que el nudo actual se cederá el control a la instrucción 12300. Si es "menor" que el almacenado en el nudo, se tendrá:

```
12110 IF CVI(SI$)<>0 THEN P%=CVI(SI$):GOTO 12500
```

Si el puntero izquierdo contiene un valor diferente de 0, obligamos a que P% tome el valor del puntero izquierdo, y si HECHO sigue siendo igual a 0, volvemos a recorrer el árbol; si no es así, se tendrá:

```
12130 LSET SI$=MKI$(ACTUAL%)
12140 GOSUB 16500:REM -->ESCRITURA INDICE
```

Actualizamos el nudo apuntado por P% y lo escribimos en disco, con lo que se tendrá:

```
12145 NUMERO%=ACTUAL%
12146 GOSUB 16000:REM -->LECTURA INDICE
12147 LSET NO$=NOMINATIVO$
```

```
12150 LSET SI$=MKI$(0)
12160 LSET DE$=MKI$(-P%)
12165 GOSUB 16500:REM -->ESCRITURA INDICE
12510 IF HECHO=0 THEN GOTO 12080
12520 REM
12530 ACTUAL%=ACTUAL%+1
12540 RETURN
```

Actualizamos, después de haberlo leído a partir del disco, el nudo apuntado por ACTUAL% (observe cómo se carga el puntero derecho). Luego se vuelve a escribir el registro en disco, se actualiza el contador de los registros y se realiza una nueva entrada de la subrutina.

Nos queda todavía examinar el caso en el que el "nominativo" tenga un valor mayor que el nudo recorrido, y para ello tendremos que examinar las instrucciones 12300/12500 del apéndice.

```
12310 IF CVI(DE$)>0 THEN P%=CVI(DE$):GOTO 12500
```

Si el puntero derecho contiene un valor mayor que 0, llevamos a P% el puntero derecho y recorremos el nudo con una iteración del ciclo o de otro modo:

```
12330 M%=CVI(DE$)
12340 LSET DE$=MKI$(ACTUAL%)
12350 GOSUB 16500
```

Actualizamos y escribimos en disco el nudo apuntado por P%, aunque no antes de haber almacenado en M% el puntero derecho de dicho nudo, y luego:

```
12355 NUMERO%=ACTUAL%
12356 GOSUB 16000:REM -->LECTURA INDICE
12360 LSET NO$=NOMINATIVO$
12365 LSET DE$=MKI$(M%)
12370 LSET SI$=MKI$(0)
12380 GOSUB 16500:REM -->ESCRITURA INDICE
```

Actualizamos el nudo apuntado por ACTUAL% después de haberlo leído a partir del disco y lo volvemos a escribir después en disco.

El procedimiento ACT_NUDO está incluido en el programa IS_GESTION del apéndice reservado al lenguaje Pascal. Como es habitual, pasamos a comentar las instrucciones principales:


```
hecho:=FALSE;
IF actual=2
THEN BEGIN   {si primer registro}
 nombre:=nominativo;
 izquierdo:=0;
 derecho:=0;
 SEEK(indice,2);
 PUT(indice);
END
```

En el caso de que se tenga que grabar el primer nudo, es decir, la raíz del árbol, se pondrán a 0 los dos punteros, se escribirá el registro y se hará una nueva entrada de la subrutina; de no ser así:

```
ELSE BEGIN
   p:=2;
   REPEAT
      SEEK(indice,p);
      GET(indice);
      IF nominativo>cliente
      THEN BEGIN
```

Después de haber cargado la variable "P" con el valor 2, se da comienzo a un bucle de instrucciones para la búsqueda del nudo en donde insertar el nuevo "nominativo". En el caso de que este último sea "mayor" que el nudo actual, se ejecutan las instrucciones siguientes:

```
IF derecho>0
  THEN p:=derecho
```

Si el puntero derecho contiene un valor mayor que 0 llevamos a "P" este valor y leemos el nudo con una nueva entrada en el ciclo o, de otro modo:

```
ELSE BEGIN
    hecho:=TRUE;
    m:=derecho;
    derecho:=actual;
    SEEK(indice,p);
    PUT(indice);
```

Actualizamos y escribimos en el disco el nudo apuntado por "P", no antes de haber almacenado en "M" el puntero derecho de ese nudo y luego:

```
        SEEK(indice,actual);
        derecho:=m;
        izquierdo:=0;
        nombre:=nominativo;
        PUT(indice);
     END;
  END
```

Actualizamos el nudo apuntado por ACTUAL.

Nos queda por examinar el caso en el que el "nominativo" tenga un valor menor que el nudo recorrido:

```
  ELSE BEGIN
  IF izquierdo<>0
    THEN p:=izquierdo;
```

Si el puntero izquierdo contiene un valor diferente de 0, llevamos a "P" este valor y volvemos a recorrer el árbol:

```
  ELSE BEGIN
      hecho:=TRUE;
      izquierdo:=actual;
      SEEK(indice,p);
      PUT(indice);
```

Actualizamos el nudo apuntado por "P" y lo escribimos en disco y luego:

```
        SEEK(indice,actual);
        izquierdo:=0;
        derecho:=-p;
        nombre:=nominativo;
        PUT(indice);
     END;
  END;
```

Actualizamos el nudo apuntado por "P"; observe cómo se carga el puntero derecho.

```
  UNTIL hecho;
```


Se saldrá del bucle solamente si la variable HECHO se pone en VERDADERO.

```
    actual:=actual+1;
  END;
```

Se actualiza el contador de registros y de salidas de la subrutina.

El programa CLAVE tiene también un funcionamiento sencillo: si se encuenta el "nominativo", se pondrá a "verdadero" la variable booleana ENC y se toma de PROXIMO el nudo que sigue; finalmente se realiza una nueva entrada de la subrutina. De otro modo:

```
    ELSE IF derecho>0 THEN j:=derecho
                       ELSE j:=0;
  END;
```

Si el "nominativo" a buscar es mayor que el existente en el nudo examinaremos el siguiente nudo solamente si el puntero derecho contiene un valor mayor que 0 y, de no ser así, se sale de la subrutina.

Se sale del bucle de instrucciones en el caso de que las variables ENC o FIN estén puestas a "verdadero". En el caso de que se encuentre el "nominativo", se leerá el registro para ponerlo a disposición del usuario.

Subrutina LECTURA

Permite leer un cierto número de registros y visualizarlos en la pantalla. Se piden los números del primer y del último registros a leer. El procedimiento LISTA tiene como objeto encontrar el enésimo registro, a condición de que se haya grabado. Si el registro no existe, se activará la variable FIN.

Se sale de la subrutina cuando se pulsa la tecla <Escape>, cuando se ha visualizado el último registro solicitado o cuando se han visualizado todos los registros existentes en el fichero.

Esta subrutina se comentó ya en el volumen anterior.

Subrutina LISTA

Sirve para leer el enésimo registro con tal de que esté grabado. Si el registro no se ha grabado (cuando INICIO>=ACTUAL e INICIO<2) se activará la variable FIN.

Subrutina NOMBRE

Permite leer los clientes mediante el campo de clave, o sea, con el nombre. La subrutina, por medio de la subrutina TECLADO, solicita desde qué nombre hasta cuál otro queremos leer. A continuación el control se cederá a la subrutina CLAVE, que se encarga de buscar el primer "nominativo".

En el caso de búsqueda de claves parciales, por ejemplo partiendo de las claves que comienzan con "b", al no encontrar la clave se activará la variable FIN y se cederá el control a la subrutina BUSQUEDA, que se posiciona en el último registro encontrado. Se entra luego en un bucle en el que se visualiza el registro CLIENTES y se cede el control a la subrutina BUSQUEDA que se encarga de buscar el siguiente registro. Se sale de la subrutina cuando se pulsa la tecla <Escape>, cuando se visualiza el último "nominativo" o cuando se visualizaron todos los registros existentes en el fichero.

La subrutina está incluida desde la instrucción 6000 a la instrucción 6540 del programa IS_GESTION en el apéndice reservado al lenguaje BASIC. Veamos las instrucciones principales:

```
6090 PRINT "NOMBRE DEL PRIMER CLIENTE:"
6100 SW=3
6110 XPOS=32
6120 YPOS=20
6130 GOSUB 1000:REM --->TECLADO
6140 PARTIDA$=ALFA$
```

Mediante la subrutina TECLADO se solicita el nombre del primer cliente para almacenarlo en la variable PARTIDA$. De forma análoga se solicita el nombre del último cliente a visualizar y se almacena en la variable TERMINO$.

```
6235 NOMINATIVO$=PARTIDA$
6240 GOSUB 13000:REM -->CLAVE
6250 IF FIN=1 THEN GOTO 6500
```

Con la ayuda de la subrutina CLAVE buscamos el "nominativo"; si no lo encontramos cedemos el control a la subrutina BUSQUEDA que nos permite hallar el nudo anterior. Si se encuentra el "nominativo", se tendrá:

```
6260 GOSUB 500:REM ---->BORRADO PANTALLA
6270 GOSUB 10000:REM -->MASCARA
6280 GOSUB 11000:REM -->VISUALIZACION
```


Se visualiza el registro en la pantalla.

```
6287 GOSUB 13500:REM -->CERCA
6290 YPOS=20:XPOS=1
6300 GOSUB 15800:REM -->POSICIONA EL CURSOR
6310 PRINT "<ESPACIO> PARA SEGUIR,<ESC>
     PARA TERMINAR:"
6320 GOSUB 15900:REM -->LECTURA CARACTER
```

Se lee el siguiente registro y luego se da la posibilidad al usuario de observar la pantalla y decidir si se verán los siguientes registros o si se produce una nueva entrada en la subrutina.

El procedimiento PARA_NOMBRE se incluye en el programa IS_GESTION en el apéndice destinado al lenguaje Pascal. Veamos las instrucciones principales:

```
GOTOXY(0,20);
WRITE('nombre del primer cliente  :');
sw:='3';
x_pos:=32;
y_pos:=20;
teclado;
partida:=alfa;
```

Mediante el procedimiento TECLADO se solicita el nombre del primer cliente para almacenarlo en la variable PARTIDA. De forma análoga se solicita el nombre del último cliente a visualizar y se almacena en la variable TERMINO.

```
nominativo:=partida;
clave;
cual:=proximo;
IF fin
THEN BEGIN
 busqueda;
 fin:=FALSE;
END;
```

Con la ayuda del procedimiento CLAVE buscamos el "nominativo" y si no lo encontramos cedemos el control al procedimiento BUSQUEDA, que nos permite volver al nudo anterior. Si se encuentra el "nominativo":

```
REPEAT
 borrado_pantalla;
 mascara;
 visualizacion;
```

Se visualiza el registro en la pantalla.

```
busqueda;
WRITE('<ESPACIO> para continuar,',
     '<ESC> para terminar');
READ(respuesta);
```

Se lee el siguiente registro dando al usuario la posibilidad de observar la pantalla y luego decidir si quiere ver los registros siguientes o acabar y entrar de nuevo en la subrutina.

```
UNTIL(respuesta=CHR$(27))
   OR(presente=0)
   OR(indice^.nombre>termino)
```

El bucle termina por indicación del usuario (cuando pulsa la tecla <Escape>), cuando se ha visualizado el último cliente solicitado o, finalmente, cuando se han acabado los registros en el fichero de clientes.

Subrutina BUSQUEDA

Esta subrutina lee los registros del fichero INDICE y del fichero CLIENTES que corresponden al contenido de la variable CUAL. En caso de que el puntero izquierdo sea diferente de 0 se continuará la búsqueda basándose en su contenido. La variable CUAL, al final de la subrutina, se fija de modo que apunte al registro siguiente.

La subrutina BUSQUEDA está incluida en el apéndice destinado al lenguaje BASIC, incorporada en el programa IS_GESTION, del que ocupa las instrucciones 13500/13820.

```
13540 FUERA=0
13550 REM
13560 ESTO%=ABS(PROXIMO%)
13565 NUMERO%=ESTO%
13570 GOSUB 15000:REM -->LECTURA CLIENTE
```


Después de haber inicializado la variable FUERA se lee un registro del fichero INDICE. El número del registro leído depende del valor absoluto contenido en la variable PROXIMO%, porque puede contener también valores negativos.

```
13580 IF CVI(SI$)<>0 AND PROXIMO%>=0 THEN
      PROXIMO%=CVI(SI$):GOTO 13800
```

Se almacena el siguiente nudo a condición de que el puntero izquierdo no esté puesto a 0 y de que la variable PROXIMO% no contenga un valor negativo; luego se vuelve a efectuar la lectura, o de otro modo:

```
13590 FUERA=1
13600 PROXIMO%=CVI(DE$)
```

Se almacena el siguiente nudo y se realiza una nueva entrada en la subrutina.

El procedimiento BUSQUEDA está incluido en el programa IS_GESTION del apéndice destinado al lenguaje Pascal. Veamos las instrucciones principales:

```
fuera:=FALSE;
REPEAT
  presente:=ABS(cual);
  SEEK(indice,presente);
  GET(indice);
  SEEK(cliente,presente);
  GET(cliente);
```

Después de haber inicializado la variable FUERA se comienza un bucle de instrucciones en donde se lee un registro de INDICE y el registro CLIENTES correspondiente. El número del registro depende del valor absoluto de la variable PROXIMO, ya que puede contener valores negativos:

```
IF(izquierdo<>0)
AND (cual)=0)
THEN cual:=izquierdo
```

Se lee el siguiente nudo a condición de que el puntero izquierdo contenga un valor diferente de 0 y la variable PROXIMO contenga un valor mayor o igual a 0.

```
    ELSE BEGIN
      fuera:=TRUE;
      cual:=derecho;
    END;
  UNTIL fuera;
```

Se halla el siguiente nudo y se sale del procedimiento. Hay que destacar el hecho de que se sale del procedimiento solamente cuando la variable booleana FUERA está puesta a "Verdadero".

Subrutina MODIFICACION

El acceso mediante clave nos permite modificar cualquier registro. Después de haber borrado la pantalla se solicita, mediante la subrutina TECLADO, el nombre del cliente a modificar. La subrutina CLAVES se encarga de buscar dicho "nominativo" en el fichero INDICE. Si no se encuentra el "nominativo" se da la indicación correspondiente en la pantalla y se sale de la subrutina. Si encontró el "nominativo" será posible modificar todos los campos a excepción del correspondiente al nombre del cliente. El cursor se posiciona al comienzo del campo; si el operador pulsa la tecla <Escape> podrá modificarlo (escribirá los datos nuevos y concluirá con <Return>). Finalmente, el registro se volverá a escribir en el disco exactamente en la misma posición.

Debería resultar intuitivo que, al no poder modificar el campo NOMBRE, el fichero índice se actualizará con el contenido del campo de nombre y luego, al modificarlo, no nos será posible volverlo a encontrar. Si, por ejemplo, modificamos el nombre "casa" en "casado", no lograremos volver a encontrar "casado". Alguno de nuestros lectores se preguntará qué tendrá que hacer si se equivoca en la clave durante la carga de los datos. La respuesta es que tendrá que suprimir la clave "casa" e insertar la clave "casado" escribiendo también los demás datos.

Subrutina BORRADO

Anteriormente dijimos que en nuestro propósito de no complicar los programas no hemos utilizado la técnica de recuperar inmediatamente el espacio libre dejado por los borrados; será precisamente el programa IS_COMPACTA el encargado de recuperar el espacio. Después del borrado de la pantalla se pedirá la clave de búsqueda mediante la subrutina TECLADO.

El control se cederá a la subrutina CLAVE. Si no se encontrara el nombre se dará la indicación correspondiente en la pantalla y se saldrá de la subrutina.

Si se encuentra la clave se visualizará el registro en la pantalla. Luego se solicita la confirmación de borrado al usuario. Si este último responde con "S" o con "s", el campo TIPO se modifica en "C" y luego el registro se graba en disco en la misma posición.

Subrutina LECTURA_INDICE

Sirve para leer de manera aleatoria un registro del fichero INDICE.

Subrutina ESCRITURA_INDICE

Escribe, con la ayuda de la variable NUMERO%, un registro en el fichero INDICE.

Programa IS_COMPACTA

Antes de utilizar este programa es necesario hacer algunas operaciones preliminares: cambiar el nombre del fichero CLIENTES.DAT por CLIENTES.VIE y llamar al programa IS_PREPARA que establece el nuevo fichero CLIENTES.DAT.

En BASIC haremos RENAME CLIENTES.DAT CLIENTES.VIE. En Pascal utilizaremos la función C)ambio del F)ichero y luego escribiremos CLIENTES.DAT, CLIENTES.VIE. Luego llamaremos a IS_PREPARA para establecer el nuevo fichero CLIENTES.DAT y el fichero INDICE.DAT. Durante esta fase se pueden dimensionar los ficheros de manera diferente.

El programa está incluido en el apéndice destinado al lenguaje BASIC con el nombre de IS_COMPACTA. Veamos las instrucciones principales:

```
140 OPEN "R",2,"CLIENTES.DAT",88
150 OPEN "R",1,"CLIENTES.VEC",88
```

Se abren los dos ficheros declarándolos de tipo aleatorio.

```
175 GET #2,1
```

Se lee el primer registro del fichero VIEJOS para conocer el número de los registros allí grabados y el primer registro del fi-

chero CLIENTES para conocer cuántos registros puede contener el fichero.

```
200 OPEN "R",3,"INDICE.DAT",35
```

Se abre el fichero INDICE declarándolo de tipo aleatorio.

```
210 X%=CVI(ULTIMO$)
220 Y%=CVI(ATC$)
```

Almacenamos en X% el número de registros que puede contener el fichero y en Y% el número del primer registro a grabar.

```
240 ACTUAL%=2
```

Después de haber inicializado las variables ACTUAL% y Z%, iniciamos un bucle de instrucciones que termina después de haber leído el último registro del fichero VIEJOS.

```
280 GET #1,Z%
290 IF TIF$<>"C" THEN GOTO 320
```

Se lee un registro del fichero VIEJOS y se lleva la indicación de borrado a visualizarse en la pantalla, se incrementa la variable Z% y se cede el control a la instrucción 480. De otro modo:

```
330 IF Z%>X$+1 THEN GOTO 500
```

Si no hay espacio en disco se da la indicación y se bloquea el programa; de otro modo:

```
360 LSET CL$=CLI$
```

Se llevan al buffer del fichero CLIENTES los diversos campos utilizando instrucciones similares a la indicada.

```
450 PRINT CL$
460 NUMERO%=ACTUAL%
465 GOSUB 15500:REM -->ESCRITURA CLIENTE
470 GOSUB 12000:REM -->ACT_NUDO
```

Se graba el fichero CLIENTES y se llama la subrutina ACT_NUDO que se comentó ya en el programa IS_GESTION.

```
475 Z%=Z%+1
480 IF Z%<Y%+1 THEN GOTO 260
```


Si hay otros registros, se cede el control a la instrucción 260; de otro modo:

```
620 NUMERO%=1
625 GOSUB 15000:REM -->LECTURA CLIENTE
630 LSET ACT$=MKI$(ACTUAL%)
640 GOSUB 15500:REM -->ESCRITURA CLIENTE
```

Grabamos la situación de los registros válidos en el primer registro del fichero CLIENTES.

```
650 CLOSE 1
660 CLOSE 2
670 CLOSE 3
```

Cerramos los tres ficheros.

En el apéndice destinado al lenguaje Pascal hemos incluido el programa IS_COMPACTA. Destacamos las instrucciones principales:

```
RESET(viejo,'cliente.vie');
RESET(indice,'indice.dat');
RESET(cliente,'cliente.dat');
```

Se abren los tres ficheros.

```
SEEK(cliente,1);
GET(cliente);
SEEK(viejo,1);
GET(viejo);
ultimo:=cliente^.maximo;
y:=viejo^.primero;
```

Se lee el primer registro del fichero de clientes para conocer el número de los registros allí grabados. Se almacena en la variable ULTIMO el número máximo de registros efectuando la lectura correspondiente desde el primer registro del fichero CLIENTES. Luego, después de haber inicializado las variables ACTUAL y Z, se da comienzo a una serie de instrucciones que terminan cuando se ha leído el último registro del fichero VIEJOS.

```
GET(viejo);
actual:=2;
z:=2;
```

```
REPEAT
 WITH viejo^
 DO BEGIN
       IF tipo='C'
       THEN WRITELN(cliente:15,' borrado')
```

Se lee un registro del fichero VIEJOS; si este último lleva la indicación de borrado se señalará en la pantalla. De otro modo:

```
ELSE BEGIN
  WRITELN(cliente:15);
  cliente^:=viejo^;
  SEEK(cliente,actual);
  PUT(cliente);
  nominativo:=viejo^.cliente;
  act_nudo;
END;
```

En la pantalla se visualiza el nombre del cliente y luego se cede el control al procedimiento ACT_NUDO, que comentamos con anterioridad.

```
IF actual-1>ultimo
THEN BEGIN
  WRITELN('hay',actual-1,' registros',
       'activos en un fichero que los',
       'prevee ',ultimo,'!');
  WRITELN('no puedo seguir.');
  WRITELN('pulse <RETURN>');
  READLN;
  EXIT(program);
END;
```

Si no hay espacio para grabar otros registros se da la indicación correspondiente en la pantalla y se bloquea la continuación del programa; de otro modo:

```
        GET(viejo);
        z:=z+1;
      END;  {de la with viejo}
UNTIL z>=y;
```


Se lee el registro siguiente y si hay otros registros a procesar, se vuelve al bucle; de otro modo:

```
SEEK(cliente,1);
GET(cliente);
cliente^.primo:=actual;
SEEK(cliente,1);
PUT(cliente);
```

Se actualiza el primer registro del fichero CLIENTES para recordar el siguiente registro a grabar.

```
CLOSE(viejo);
CLOSE(indice);
CLOSE(cliente);
```

Cerramos los tres ficheros.

Después de los oportunos controles será posible borrar el fichero CLIENTES.VIE. En el lenguaje BASIC haremos ERASE CLIENTES.VIE; llamaremos a la función R)etirada del F)ichero, escribiremos CLIENTES.VIE y responderemos "Y" a la solicitud de confirmación del borrado.

CAPITULO II

ACCESO HASH

Esta organización nos permite el acceso a cualquier registro, tanto mediante una clave como con el número de registro. No es posible realizar una búsqueda en clave parcial. Las claves están almacenadas en un fichero que contiene solamente punteros.

Describiremos solamente tres programas: el primero, HASH_PREPARA, establece los dos ficheros; el segundo, HASH_GESTION, nos permite cargar los datos, modificarlos, borrarlos o leerlos, y el último programa, HASH_COMPACTA, vuelve a crear los dos ficheros eliminando los registros borrados. La disposición del registro tipo del fichero CLIENTES es como sigue:

CLIENTE	campo alfabético de 31 caracteres
DIRECCION	campo alfabético de 33 caracteres
CIUDAD	campo alfabético de 15 caracteres
TELEFONO	campo numérico de 8 caracteres
PREFIJO	campo alfabético de 4 caracteres
TIPO	campo alfabético de 1 carácter
COLISION	campo numérico de 2 bytes

El primer registro del fichero CLIENTES contiene dos informaciones: la primera indica el número del primer registro que debe escribirse y la segunda indica cuántos registros se pueden grabar en el fichero. Los registros del fichero INDICE contienen un solo campo, de tipo numérico, que indica en dónde se encuen-

tran los datos del cliente que tiene el número hash igual al número de dicho registro.

Programa HASH_PREPARA

Este programa pide cuantos registros, como máximo, se escribirán en el fichero.

No siempre sabemos con exactitud dicho valor. Podemos hacer una estimación y aumentarla en un 20 por 100 por precaución. Si en la utilización nos percatamos de haber superado las previsiones, podemos generar un fichero más amplio siguiendo las instrucciones que se darán en la explicación del programa HASH_COMPACTA. El programa almacena en el primer registro dos informaciones. En el primer campo se escribe un 2; de hecho, la primera información se escribirá en el segundo registro. En el segundo campo incluiremos el número de registros que puede contener el fichero.

Luego se escriben todos los registros posibles con los campos alfabéticos vacíos (a excepción del prefijo, que se pone en 0000) y los campos numéricos a 0, mientras que el campo COLISION se pone a -1. Luego se escribe el fichero INDICE; todos sus registros contienen el valor -1. El número de los registros será igual al valor de DIVISOR, que podrá variarse según las dimensiones del fichero. El programa en BASIC HASH_PREPARA está incluido en el apéndice dedicado al lenguaje BASIC. Veamos las instrucciones principales:

```
140 FICHERO$="CLIENTE.DAT"
150 DIVISOR=7
160 OPEN "R",1,FICHERO$,90
```

Hemos considerado que 7 es el valor a asignar a la variable DIVISOR. A continuación se abre el fichero CLIENTES.DAT declarándolo de tipo "ALEATORIO" y determinando que los registros contendrán, como máximo, 90 caracteres.

```
170 INPUT "¿DE CUANTOS REGISTROS ESTARA COMPUESTO
    EL FICHERO?";ULTIMO$
180 PRIMERO=2
190 FIELD 1,2 AS PRIMERO$,2 AS ULTIMO$
```

Después de haber solicitado al operador cuántos podrán ser los registros del fichero, indicamos las características del primer

registro mediante la instrucción FIELD; en este caso, estará constituido por dos campos, cada uno de ellos de dos caracteres.

```
200 LSET PRIMERO$=MKI$(PRIMERO)
210 LSET ULTIMO$=MKI$(ULTIMO+1)
```

Mediante estas instrucciones transferimos al buffer, posición de memoria reservada para admitir los datos de los ficheros, los contenidos de las variables PRIMERO y ULTIMO. Son necesarias las instrucciones LSET y MKI$ para transferir al buffer un dato numérico en el formato de cadena.

```
220 PUT #1,1
```

Escribimos el primer registro del fichero.

```
230 FIELD 1,31 AS CLIENTE$,33 AS DIRECCION$,15 AS
    CIUDAD,4 AS TELEFONO$,4 AS PREFIJO,1 AS COLISION$
```

Declaramos las características de los sucesivos registros del fichero CLIENTES.

```
250 LSET CLIENTE$=""
```

Dejamos vacío el espacio destinado a admitir el campo CLIENTE; luego siguen otras instrucciones para inicializar adecuadamente los demás campos. Recordemos también que en el acceso aleatorio se tiene que utilizar la instrucción LSET para transferir los datos a los campos del registro.

```
320 FOR X%=2 TO ULTIMO+1
330 PUT #2,X%
340 NEXT X%
```

Se activa un bucle de instrucciones que graba los registros declarados por el usuario, estando cada campo del registro adecuadamente inicializado. Después de haber cerrado el fichero CLIENTES se abre el fichero INDICE.DAT, en donde se grabarán los registros, todos ellos puestos a -1.

```
410 FICHERO$="INDICE.DAT"
420 OPEN "R",2,FICHERO$,2
430 FIELD 2,2 AS INDICE$
```

Se abre un fichero de tipo aleatorio constituido por un solo campo de dos caracteres.

```
435 INDICE%=-1
440 LSET INDICE$=MKI$(INDICE%)
```

El único campo del registro se pone al valor -1.

```
450 FOR X=1 TO DIVISOR
460 PUT #2,2
470 NEXT X
```

En el fichero se graba un número de registros igual al valor contenido en la variable DIVISOR; en nuestro ejemplo 7 registros. A continuación, se cerrará el fichero.

El programa HASH_PREPARA está incluido en el apéndice destinado al lenguaje Pascal. Pondremos de manifiesto las instrucciones principales.

```
REWRITE(cliente,'cliente.dat');
WRITE('¿de cuantos registros estara compuesto',
      'el fichero?');
READLN(ultimo);
```

Después de haber abierto el fichero se pregunta al usuario cuántos serán los registros del fichero para su almacenamiento en la variable ULTIMO. El lenguaje Pascal tiene la posibilidad de declarar diversos tipos de registros en el mismo fichero con el empleo de la palabra clave CASE en la declaración del registro; hay que observar, a tal propósito, la declaración del tipo "CLI" en el programa HASH_PREPARA antes citado.

```
primero:=2;
maximo:=ultimo+1;
PUT(cliente);
PUT(cliente);
```

Los dos campos del registro se inicializan, respectivamente, con 2 y con el valor proporcionado por el usuario; mediante PUT se graban los registros en disco.

```
cliente:='';
direccion:='';
ciudad:='';
telefono:=0;
prefijo:='0000';
tipo:=' ';
colision:=-1;
```


Todos los campos se ponen a 0 o están vacíos y, en particular, el campo de colisión se pone a -1.

```
FOR x:=2 TO ultimo+1 DO PUT(cliente);
```

Se graba un número de registros igual a los declarados por el usuario. Después del cierre del fichero hay que seguir las instrucciones que se indican a continuación.

```
REWRITE(indice,'indice.dat');
indice^:=-1;
FOR x:=0 TO divisor-1 DO PUT(indice);
CLOSE(indice,LOCK);
```

Se abre el fichero INDICE, que contendrá un número de registros igual al contenido de la variable DIVISOR; cada registro está constituido por un solo campo que contiene el valor -1.

Programa HASH_GESTION

Este programa se ha realizado siguiendo algunas normas de la programación estructurada, subdividiéndolo en numerosas subrutinas. Se ha reducido al mínimo el núcleo fundamental del programa. Un menú suficientemente explicativo permite la elección entre diversas opciones, haciendo uso cada opción de varias subrutinas.

El programa principal sirve para abrir el fichero INDICE y el fichero CLIENTES y para almacenar, en la variable ACTUAL, el número del primer registro a grabar, y en la variable ULTIMO el número de los registros existentes en el fichero. La variable DIVISOR es determinante para el cálculo del número hash y se colocó al comienzo del programa para que se pueda identificar con facilidad y luego modificarse. El programa está incluido por completo, tanto en el apéndice destinado al lenguaje BASIC como en el apéndice del Pascal, con el nombre de HASH_GESTION.

Comentemos las instrucciones principales en BASIC.

```
140 OPEN "R",1,"CLIENTES.DAT",90
150 OPEN "R",2,"INDICE.DAT",2
160 DIVISOR=7
170 NUMERO%=1
180 FIELD 1,2 AS ACTUAL$,2 AS ULTIMO$
190 GOSUB 15000:REM -->LECTURA CLIENTE/1
```

Después de haber abierto los dos ficheros se lee el primer registro del fichero CLIENTES para almacenar el primer registro a grabar y el número de los registros que constituyen el fichero.

```
200 ACTUAL%=CVI(ACTUAL$)
210 ULTIMO%=CVI(ULTIMO$)
```

Las dos variables ACTUAL% y ULTIMO% nos indican, respectivamente, cuál será el primer registro a grabar y cuántos registros contiene el fichero. Para transformar una variable de cadena en una variable numérica se utiliza CVI.

```
230 GOSUB 800:REM ---->MENU
240 ON QUEQUIERE GOSUB 2000,4000,6000,8000,9000
250 IF QUEQUIERE<>9 GOTO 220
```

La instrucción 240 cede el control a la subrutina que se inicia con la instrucción 2000 si la variable QUEQUIERE contiene el valor 1, con la instrucción 4000 si dicha variable contiene el valor 2 y así sucesivamente. En el caso de que QUEQUIERE contenga un valor mayor que 5, el control se cederá a la instrucción 220 si el valor es diferente de 9 y, de no ser así, el programa se cerrará.

```
270 CLOSE 1
280 CLOSE 2
290 END
```

Veamos las instrucciones principales del programa en Pascal.

```
RESET(indice,'indice.dat');
RESET(cliente,'cliente.dat');
SEEK(cliente,1);
GET(cliente);
actual:=cliente^.primero;
ultimo:=cliente^.maximo;
```

Después de la apertura de los dos ficheros se almacenan las informaciones relativas al primer registro a grabar y al número de los registros del fichero.

```
REPEAT
  menu;
  CASE que_quiere OF
       '1':carga;
       '2':lectura;
       '3':nombre;
       '4':modificacion;
       '5':borrado;
     END;
  UNTIL que_quiere='9';
  CLOSE (indice.LOCK);
  CLOSE(cliente.LOCK);
```

Consideramos estas instrucciones tan evidentes que creemos innecesario comentarlas.

Subrutina CARGA

Después de borrar la pantalla se visualizará el número del primer registro a grabar. Ello puede ser interesante cuando al reanudar la carga queramos saber cuántos registros hemos cargado ya y, luego, valorar cuánto se empleará para cargar el resto. Si se está en una fase de carga avanzada será útil saber si la operación se realizó de forma correcta, con la visualización del último registro grabado.

Subrutina CALCULO

Esta subrutina determina el número Hash del contenido de la variable NOMINATIVO.

Todos los caracteres de la variable se convierten en su valor binario. Los valores binarios de los caracteres que ocupan la posición impar en la variable NOMINATIVO se añaden en la variable "X", mientras que los demás se suman en la variable "Y". En R1 pondremos el resto de la división de la variable "X" por 256 y en la variable R2 el resto de la división de la variable "Y" por 256. El contenido de la variable "Z" viene determinado por la fórmula (R2*256)+R1.

La variable HASH contendrá el resto de la división de la variable "Z" por el valor de DIVISOR. Si constatamos que hay demasiadas colisiones aumentaremos el valor de DIVISOR, tratando de utilizar un número primo.

La subrutina en BASIC está incluida en el apéndice correspondiente, en el programa HASH_GESTION, desde la instrucción 12000 a la instrucción 12140.

```
12040 X=0:Y=0
12050 FOR N=1 TO LEN(NOMINATIVO$)
```

Después de poner a 0 las variables "X" e "Y" se da comienzo a un bucle en donde se examina cada carácter individual contenido en la variable NOMINATIVO$

```
12060 C=ASC(MID$(NOMINATIVO$,N,1))
```

En "C" tenemos el valor binario del enésimo carácter de la variable NOMINATIVO$.

```
12070 IF N-(INT(N/2)*2)=0 THEN Y=Y+C ELSE X=X+C
```

Si examinamos las posiciones pares de la variable NOMINATIVO$, la variable "C" se sumará a la variable "Y" y, en caso contrario, se sumará a la variable "X".

```
12080 NEXT N
```

Se desactiva el bucle cuando se han examinado todos los caracteres de la variable.

```
12090 R1=X-(INT(X/256)*256)
12100 R2=Y-(INT(Y/256)*256)
```

La variable R1 contiene el resto de la división de "X" por 256, mientras que la variable R2 contiene el resto de la división de "Y" por 256.

```
12110 Z=R2*256+R1
12120 HASH=INT(Z-(INT(Z/DIVISOR)*DIVISOR))
12130 IF HASH=0 THEN HASH=DIVISOR
```

La variable HASH contiene el resto de la división de "Z" por el valor de DIVISOR y, en el caso de que el valor sea 0, la variable HASH tomará el valor contenido en la variable DIVISOR.

El procedimiento CALCULO está incluido por completo en el apéndice dedicado al lenguaje Pascal, en el programa HASH_GESTION. Nos limitaremos a comentar las instrucciones principales:

```
x:=0;
y:=0;
FOR n:=1 TO LENGTH(nominativo)
DO BEGIN
```


Las variables "X" e "Y" se ponen a 0 y luego se inicia una serie de instrucciones que se repetirá un número de veces igual al número de caracteres contenidos en la variable NOMINATIVO.

```
c:=ORD(nominativo[n]);
IF ODD(n) THEN x:=x+c
        ELSE y:y+c;
```

Cada carácter individual se transforma en su valor binario y se suma a la variable "X" si se examina un carácter que ocupa posiciones pares de la variable NOMINATIVO y, de no ser así, se suma a la variable "Y". Después de haber examinado todos los caracteres de la variable NOMINATIVO se tendrá:

```
r1:=x-((x DIV 256)*256);
r2:=y-((y DIV 256)*256);
z:=r2*256+r1
```

En R1 se almacena el módulo 256 de la variable "X" y en R2 el módulo 256 de la variable "Y", mientras que en la variable "Z" tenemos el valor de la variable R2 multiplicada por 256 más el valor de la variable R1.

```
hash:=TRUNC(z-((z DIV divisor)*divisor));
IF hash=0 THEN hash:=divisor;
```

La variable "Z" se divide por el valor contenido en la variable DIVISOR y el resto de la división se almacena en la variable HASH; en el caso de que esta última contenga el valor 0 se tomará el valor contenido en la variable DIVISOR.

Subrutina BUSQUEDA

La subrutina BUSQUEDA en BASIC está incluida en el apéndice desde la línea 13000 a la línea 13140 del programa HASH_GESTION. Veamos las instrucciones principales:

```
13040 FIN=0
13050 GOSUB 15200:REM -->LECTURA HASH
13060 CUAL%=CVI(HASH$)
13070 REM
13080 IF CUAL%=-1 THEN FIN=1:RETURN
```

Leemos el fichero INDICE mediante el contenido de la variable HASH. Si el registro contiene el valor -1, ello significa que no existe dicho "nominativo" y se realiza una nueva entrada en la subrutina poniendo a 1 la variable FIN. De otro modo:

```
13090 NUMERO%=CUAL%
13100 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,
      4 AS TE$,4 AS FR$,1 AS TI$,2 AS CO$
13110 GOSUB 15000:REM -->LECTURA CLIENTE
13120 IF NOMINATIVO$=CL$ THEN RETURN
```

Se lee el fichero CLIENTES y se compara si el nombre del cliente corresponde al almacenado en NOMINATIVO$, en cuyo caso se realiza una nueva entrada en la subrutina. Hay que destacar el hecho de que la variable FIN se mantiene en el valor 0.

```
13130 CUAL%=CVI(CO$)
13140 GOTO 13070
```

Si los valores no están en correspondencia en la variable CUAL% se sitúa el contenido de la variable CO$ y se reenvía el proceso a la instrucción 13080. El procedimiento BUSQUEDA está incluido en el apéndice dedicado al lenguaje Pascal dentro del programa HASH_GESTION.

```
fin:=FALSE;
SEEK(indice,hash);
GET(indice);
cual:=indice^;
REPEAT
 IF cual=-1
 THEN fin:=TRUE
```

Leemos el fichero INDICE mediante la variable HASH; si el registro contiene -1 ello significa que dicho nominativo no existe, por lo cual salimos del procedimiento poniendo a 1 la variable FIN; de no ser así tendremos:

```
ELSE BEGIN
     SEEK(cliente,cual);
     GET(cliente);
     IF nominativo<>cliente^.cliente
     THEN cual:=cliente^.colision;
   END;
```


Se lee el fichero CLIENTES mediante la variable CUAL y se comprueba si el nombre del cliente es igual al contenido en la variable NOMINATIVO; en tal caso se realiza un reenvío del procedimiento.

```
UNTIL fin OR (nominativo=cliente.cliente)
```

Se vuelve a leer el registro a no ser que la variable FIN esté puesta a 1 o se haya encontrado el "nominativo".

Subrutina NOMBRE

Esta subrutina solicita al usuario el nombre de un cliente, lo busca en el archivo y, finalmente, lo visualiza en la pantalla.

La subrutina está incluida por completo en el programa HASH_GESTION en el apéndice destinado al lenguaje BASIC, en las instrucciones 6000/6290. Como es nuestra costumbre, comentaremos las instrucciones principales:

```
6070 PRINT "NOMBRE DE CLIENTE     :"
6080 SW=3
6090 XPOS=32
6100 YPOS=20
6110 GOSUB 1000:REM --->TECLADO
6120 NOMINATIVO$=ALFA$
```

Se solicita el nombre del cliente mediante la subrutina TECLADO.

```
6125 GOSUB 12000:REM -->CALCULO HASH
6130 IF LEN(NOMINATIVO$)=31 THEN GOTO 6150
6140 NOMINATIVO$=NOMINATIVO$+" ":GOTO 6130
6160 GOSUB 13000:REM -->BISQUEDA
```

Se calcula el número Hash correspondiente a dicho "nominativo" y luego se cede el control a la subrutina BUSQUEDA, que se encarga de encontrar el cliente. En el caso de que no lo encuentre la variable FIN se pondrá a 1 y se avisará al usuario de que no existe dicho nombre.

```
6180 GOSUB 500:REM ---->BORRA PANTALLA
6190 GOSUB 10000:REM -->MASCARA
6200 GOSUB 11000:REM -->VISUALIZACION
```

A estas instrucciones, que no precisan ser comentadas, siguen otras para permitir al usuario observar con comodidad el registro visualizado en la pantalla.

En el apéndice destinado al lenguaje Pascal se incluyen las instrucciones correspondientes al procedimiento NOMBRE del programa HASH_GESTION. Veamos las instrucciones principales:

```
WRITE('nombre del cliente     :');
sw:='3'
x_pos:=32;
y_pos:=20;
teclado;
nominativo:=alfa;
calculo;
busqueda;
```

Se solicita mediante el teclado el nombre del cliente, luego el control se cede al procedimiento CALCULO, que tiene el objeto de calcular el número Hash y después el control se transfiere al procedimiento BUSQUEDA, que lee en el fichero el registro deseado. En el caso de que no se encuentre dicho "nominativo" se pondrá a 1 la variable booleana FIN y se dará la indicación correspondiente al usuario.

```
borrado_pantalla;
mascara;
visualizacion;
```

Tenemos la certeza de que habrá comprendido el objeto de estos tres procedimientos.

```
GOTOXY(0,20);
WRITE('pulse <RETURN> para continuar');
READLN;
```

Subrutina MODIFICACION

El acceso hash nos permite modificar cualquier registro. Después de borrar la pantalla se solicita, mediante la subrutina TECLADO, el nombre del cliente a modificar. El control se cede a la subrutina CALCULO, que determina el código Hash, y luego se llama a la subrutina BUSQUEDA. Si no se encontrara el "nominativo" se daría la indicación correspondiente en la pantalla y se saldría

de la subrutina. Si el "nominativo" fue encontrado será posible modificar todos los campos a excepción del correspondiente al nombre del cliente.

El cursor se posiciona al comienzo del campo y si el usuario pulsa la tecla <Escape> podrá modificarlo (con la escritura de los nuevos datos y concluyendo con <Return>). Finalmente, el registro se volverá a escribir en disco exactamente en la misma posición.

Esta subrutina se comentó ya con anterioridad; la encontrará incluida en el programa HASH_GESTION del apéndice. Resulta evidente por qué no se puede modificar el campo NOMBRE: el número Hash se calculó basándose en el contenido de dicho campo y si lo modificamos el número hash será diferente y, por consiguiente, ya no nos será posible encontrarlo, pues habremos realizado un direccionamiento a un registro diferente. Si, por ejemplo, modificamos el nombre "casa" por "casado", no lograremos encontrar "casado". Algunos de nuestros lectores se plantearán la interrogante de qué tendrán que hacer si escriben de forma incorrecta la clave durante la operación de carga de los datos. La respuesta es que tendrán que borrar la clave "casa" y proceder a la introducción de la clave "casado", escribiendo también las demás informaciones. Esta operación resulta elemental.

Subrutina BORRADO

Podemos marcar los registros que queremos borrar. En nuestra exposición anterior hemos dicho que para no complicar los programas no hemos empleado la técnica de recuperar inmediatamente el espacio que queda libre como consecuencia de los borrados. El programa HASH_COMPACTA se encargará de la recuperación de este espacio.

Una vez borrada la pantalla se solicita el nombre del cliente. El control se cede a la subrutina CALCULO, que determina el número Hash. Luego se pasará a la subrutina BUSQUEDA, que procederá a buscar el "nominativo" en el fichero CLIENTES. Si no se encontrara el "nominativo" se dará la indicación correspondiente en la pantalla y se saldrá de la subrutina. Si se encuentra la clave se visualiza el registro en la pantalla. Luego se solicita la confirmación del borrado al usuario. Si este último responde con "S" o "s", el campo TIPO se modificará en "C" y el registro se grabará en disco en la misma posición.

Esta subrutina, que fue comentada con anterioridad, está incluida en el programa HASH_GESTION.

Programa HASH_COMPACTA

Antes de ejecutar este programa son necesarias algunas operaciones preliminares: cambiar el nombre del fichero CLIENTES.DAT por CLIENTES.VIE y llamar al programa HASH_PREPARA, que establece el nuevo fichero CLIENTES.DAT. En el BASIC haremos RENAME CLIENTES.DAT, CLIENTES.VIE. En el lenguaje Pascal utilizaremos la función C)ambio del F)ichero y luego escribiremos CLIENTES.DAT, CLIENTES.VIE.

Se llama al programa HASH_PREPARA para establecer el nuevo fichero CLIENTES.DAT y el fichero INDICE.DAT. Durante esta fase es posible dimensionar los ficheros de manera diferente. Las dos operaciones deben realizarse en la secuencia indicada.

El programa HASH_COMPACTA abre los ficheros VIEJOS, CLIENTES e INDICE. El fichero VIEJOS contiene los registros que deben eliminarse por borrado. El fichero CLIENTES, al final del programa, contendrá solamente los registros no borrados.

El programa está incluido en el apéndice destinado al lenguaje BASIC con el nombre de HASH_COMPACTA. Veamos las instrucciones principales:

```
150 OPEN "R",1,"CLIENTE.VEC",90
160 OPEN "R",2,"INDICE.DAT",2
170 OPEN "R",3,"CLIENTES.DAT",90
```

Se abren los tres ficheros declarándolos de tipo aleatorio.

```
190 GET #1,1
200 J%=CVI(J$)
```

Se lee el primer registro del fichero VIEJOS para conocer el número de los registros allí grabados.

```
230 GET #3,1
240 ULTIMO%=CVI(ULTIMO$)
```

Del nuevo fichero CLIENTES se toma el número de registros que puede contener y dicho valor se almacena en la variable ULTIMO%. Luego, después de inicializar las variables ACTUAL%, "K", "I" y "J", iniciamos un bucle de instrucciones que termina después de haber leído el último registro del fichero VIEJOS.

```
280 GET #1,J
290 IF TI$<>" " THEN PRINT CL$;" BORRADO":J=J+1:
    GOTO 490
```


Se lee un registro del fichero VIEJOS y se lleva la indicación de borrado a la presentación visual en la pantalla, incrementándose la variable "J" y cediendo el control a la instrucción 490. De otro modo:

```
300 K=K+1:PRINT CL$
310 NOMINATIVO$=CL$
320 GOSUB 12000:REM -->CALCULO
```

Se visualiza en la pantalla el nombre del cliente, y se cede el control a la subrutina CALCULO, que determina el número Hash; esta subrutina se comentó ya en la presentación del programa HASH_GESTION y también la hemos incluido en el programa HASH_COMPACTA.

```
340 GET #2,HASH
350 HH$=HASH$
360 LSET CO$=HASH$
370 LSET HASH$=MKI$(ACTUAL%)
380 PUT #2,HASH
```

Del fichero INDICE se lee el registro que corresponde al valor contenido en la variable HASH y luego se actualiza el campo COLISION del fichero CLIENTES y también se actualiza y se graba el registro INDICE. Se actualizan los demás campos del fichero CLIENTES, volviéndolos a copiar uno a uno desde el fichero VIEJOS; se tendrá:

```
460 PUT #3,ACTUAL%
470 ACTUAL%=ACTUAL%+1
480 J=J+1
```

Se graba el fichero CLIENTES y se actualizan los contadores de registro.

```
490 REM
500 GET #1,J
510 IF ACTUAL%<ULTIMO% THEN GOTO 560
```

Se lee el registro sucesivo y, si hay todavía espacio en disco, el control se cede a la instrucción 560; de no ser así, se proporciona la indicación de dicho inconveniente y se cierra el programa.

```
560 REM
570 IF J<I% THEN GOTO 290
```

Si hay otros registros se cede el control a la instrucción 290, y de otro modo:

```
590 LSET ACTUAL$=MKI$(ACTUAL%-1)
600 LSET ULTIMO$=MKI$(ULTIMO%)
610 PUT #3,1
```

Grabamos la situación de los registros válidos en el primer registro del fichero CLIENTES.

```
630 CLOSE #1
640 CLOSE #2
650 CLOSE #3
660 END
```

Cerramos los tres ficheros.

En el apéndice destinado al lenguaje Pascal hemos incluido el programa HASH_COMPACTA. Veamos las instrucciones principales:

```
RESET(viejo,'cliente.vie');
RESET(indice,'indice.dat');
RESET(cliente,'cliente.dat');
```

Se abren los tres ficheros.

```
SEEK(viejo,1);
GET(viejo);
SEEK(cliente,1);
GET(cliente);
ultimo:=cliente^.maximo;
i:=viejo^.primero;
j:=2;
```

Se lee el primer registro del fichero CLIENTES para conocer el número de los registros allí grabados. Se almacena en la variable ULTIMO el número máximo de registros con lectura desde el primer registro del fichero CLIENTES. Luego, después de iniciar las variables ACTUAL, "K" y "J", se da comienzo a una serie de instrucciones que terminan cuando se ha leído el último registro del fichero VIEJOS.


```
GET(viejo);
actual:=2;
k:=2;
REPEAT
 WITH viejo^
 DO BEGIN
      IF tipo='C'
      THEN WRITELN(cliente:15,' borrado')
```

Se lee un registro del fichero VIEJOS; si este último lleva la indicación de borrado se señala adecuadamente en la pantalla y, de otro modo:

```
ELSE BEGIN
k:=k+1;
  WRITELN(cliente:15);
  cliente^.cliente:=cliente;
  cliente^.direccion:=direccion;
  cliente^.ciudad:=ciudad;
  cliente^.telefono:=telefono;
  cliente^.prefijo:=prefijo;
  cliente^.tipo:=' ';
  nominativo:=viejo^.cliente;
  calculo;
```

En la pantalla se visualiza el nombre del cliente y luego se cede el control al procedimiento CALCULO, que se comentó con anterioridad.

```
SEEK(indice,hash);
GET(indice);
cliente^.colision:=indice^;
indice^:=actual;
SEEK(indice,hash);
PUT(indice);
```

Del fichero INDICE se lee el registro que corresponde al contenido de la variable HASH y luego se actualiza el campo de colisión del fichero CLIENTES y se actualiza y graba INDICE. Se actualizan los demás campos del fichero CLIENTES, volviéndolos a copiar a partir del fichero VIEJOS; se tendrá:

```
    SEEK(cliente,actual);
    PUT(cliente)
    actual:=actual+1;
  END;
```

Se graba el fichero CLIENTES y se actualiza el contador de registros.

```
  j:=j+1;
  GET(viejo);
  IF actual-1>ultimo
  THEN BEGIN
```

Se lee el registro sucesivo; si todavía está en el fichero se volverá al ciclo y, de no ser así, se dará la indicación oportuna en la pantalla y se bloqueará el programa. Si no hay otros registros en el fichero VIEJOS, se tendrá:

```
  SEEK(cliente,1);
  GET(cliente);
  cliente^.primero:=actual;
  SEEK(cliente,1);
  PUT(cliente);
```

Situamos en el primer registro del fichero CLIENTES cuál será el registro siguiente a grabar.

```
  CLOSE(cliente);
  CLOSE(indice);
  CLOSE(viejo);
```

Cerramos los tres ficheros.

Después de las oportunas comprobaciones resulta posible borrar el fichero CLIENTES.VIE. En BASIC haremos ERASE CLIENTES.VIE; en Pascal llamaremos a la función R)etirada del F)ichero, escribiremos CLIENTES.VIE y responderemos con "S" a la solicitud de confirmación de borrado.

CAPITULO III

ACCESO MEDIANTE ARBOL BINARIO

Esta organización nos permite tener acceso a cualquier registro tanto por medio de una clave como por medio del número de registro. La búsqueda mediante clave puede ser también parcial. Los registros se almacenan con el empleo de la técnica del árbol binario.

Describiremoss tres programas: el primero, AL_PREPARA, establece el fichero; el segundo, AL_GESTION, nos permite cargar los datos, leerlos, modificarlos o borrarlos, y el último programa, AL_COMPACTA, vuelve a estructurar el fichero eliminando los registros borrados. La lista de campos de los registros del fichero CLIENTES es como sigue:

CLIENTE	campo alfabético de 31 caracteres
DIRECCION	campo alfabético de 33 caracteres
CIUDAD	campo alfabético de 15 caracteres
TELEFONO	campo numérico de 8 caracteres
PREFIJO	campo alfabético de 4 caracteres
TIPO	campo alfabético de 1 carácter
IZQUIERDO	campo numérico de 2 caracteres
DERECHO	campo numérico de 2 caracteres

El primer registro contiene dos informaciones: la primera indica el número del registro que debe escribirse y la segunda indica cuántos registros se pueden grabar en el fichero.

Programa AL_PREPARA

El programa pregunta cuántos registros, como máximo, se escribirán en el fichero. Si no sabemos exactamente dicho valor, podemos realizar una estimación y aumentarla un 20 por 100 por precaución.

Si luego, en la utilización, nos percatamos de haber superado las previsiones podemos generar un fichero más grande siguiendo las instrucciones que se darán en la explicación del programa AL_COMPACTA. Dicho programa tiene como objeto principal recuperar el espacio dejado libre por los borrados de registros.

El programa AL_PREPARA escribe el primer registro con dos informaciones. En el primer campo se escribe un 2 y, de hecho, la primera información se escribirá en el registro número 2. En el segundo campo incluimos el número de registros que puede contener el fichero. Luego se escriben todos los registros posibles con los campos alfabéticos vacíos (a excepción del prefijo, que se pone a 0000) y los campos numéricos que se ponen a 0.

El programa en BASIC AL_PREPARA se incluye en el apéndice reservado al lenguaje BASIC. Veamos las instrucciones principales:

```
140 FICHERO$="CLIENTES.DAT"
160 OPEN "R",1,FICHERO$,92
```

Se abre el fichero CLIENTES.DAT declarándolo de tipo aleatorio; el registro correspondiente contendrá 92 caracteres como máximo.

```
170 INPUT "¿DE CUANTOS REGISTROS ESTARA COMPUESTO
    EL FICHERO?";ULTIMO$
180 PRIMERO=2
190 FIELD 1,2 AS PRIMERO$,2 AS ULTIMO$
```

Después de haber preguntado al usuario cuántos podrán ser los registros del fichero indicamos las características del registro mediante la instrucción FIELD; en este caso, estará constituido por dos campos, teniendo cada uno de ellos dos caracteres. Este primer registro nos indica el número del siguiente registro a grabar y cuántos registros podrá contener el fichero como máximo.

```
200 LSET PRIMERO$=MKI$(PRIMERO)
210 LSET ULTIMO$=MKI$(ULTIMO)
```

Mediante estas instrucciones transferimos al buffer, posición de memoria reservada a los datos del fichero, los contenidos de las variables PRIMERO y ULTIMO. Son necesarias las instrucciones LSET y MKI$ para transferir a un registro un dato numérico en el formato de cadena.


```
220 PUT #1,1
```

Escribimos el primer registro del fichero.

```
230 FIELD 1,31 AS CLIENTE$,33 AS DIRECCION$,15 AS
    CIUDAD$,4 AS TELEFONO$,4 AS PREFIJO$,1 AS TIPO$,
    2 AS IZQUIERDO$,2 AS DERECHO$
```

Declaramos las características de los sucesivos registros del fichero CLIENTES.

```
250 LSET CLIENTE$=""
```

Dejamos vacío el espacio destinado al campo CLIENTE; siguen otras instrucciones para inicializar adecuadamente los demás campos.

```
320 FOR X%=2 TO ULTIMO+1
330 PUT #1,X%
340 NEXT X%
```

Se activa un bucle de instrucciones que graba un número de registros igual al indicado por el usuario, estando adecuadamente inicializado cada campo del registro. Finalmente se cierra el fichero CLIENTES.

El programa en Pascal AL_PREPARA está incluido en el apéndice de Pascal. Vamos a comentar las instrucciones principales.

```
REWRITE(cliente,'cliente.dat');
WRITE('¿de cuantos registros estara compuesto',
      'el fichero?');
READLN(ultimo);
```

Una vez abierto el fichero se pregunta al usuario cuántos serán los registros del fichero para su almacenamiento en la variable ULTIMO. El lenguaje Pascal tiene la posibilidad de declarar diversos tipos de registros en el mismo fichero con el empleo de la palabra clave CASE en la declaración del registro; hay que destacar, a tal propósito, la declaración del tipo "CLI" en el programa AL_PREPARA antes citado.

```
primero:=2;
maximo:=ultimo+1;
PUT(cliente);
PUT(cliente);
```

Los dos campos del registro se inicializarán, respectivamente, con 2 y con el valor proporcionado por el usuario; las instrucciones PUT grabarán los registros en disco.

```
cliente:='';
direccion:='';
ciudad:='';
telefono:=0;
prefijo:='0000';
tipo:=' ';
```

Todos los campos se quedarán vacíos o puestos a 0.

```
FOR X:=1 TO ultimo DO PUT(cliente);
```

Se graba un número de registros igual a los declarados por el usuario y el fichero se cerrará.

Programa AL_GESTION

El programa se ha realizado siguiendo algunas normas de la programación estructurada, subdividiéndolo en numerosas subrutinas. Se redujo al mínimo el núcleo fundamental del programa. Un menú suficientemente explicativo permite la elección entre varias opciones, haciendo uso cada opción de varias subrutinas. El programa principal se encarga de abrir el fichero CLIENTES y de almacenar en la variable ACTUAL el número del primer registro a grabar y en la variable ULTIMO el número de los registros existentes en el fichero.

El programa en BASIC está incluido en el apéndice correspondiente con el nombre AL_GESTION; procederemos a comentar las instrucciones principales del programa, al mismo tiempo que comentaremos las diversas subrutinas siguiendo el orden establecido en el menú.

```
140 OPEN "R",1,"CLIENTES.DAT",92
```

Abrimos el fichero CLIENTES.DAT declarándolo de acceso aleatorio y el registro correspondiente podrá contener 92 caracteres como máximo.


```
180 FIELD 1,2 AS ACTUAL$,2 AS ULTIMO$
190 GOSUB 15000:REM -->LECTURA CLIENTE/1
200 ACTUAL%=CVI(ACTUAL$)
210 ULTIMO%=CVI(ULTIMO$)
```

Se lee el primer registro para poder almacenar, en la variable ULTIMO%, el número de registros que podrá contener el fichero y en la variable ACTUAL% el número del primer registro que se puede grabar.

```
220 GOSUB 500:REM ---->BORRADO PANTALLA
230 GOSUB 800:REM ---->MENU
240 ON QUEQUIERE GOSUB 2000,4000,6000,8000,9000
```

Una vez borrada la pantalla se llama a la subrutina MENU y, basándose en la respuesta dada, se cede el control a una de las subrutinas que comienzan con la instrucción 2000, 4000, 6000, 8000 ó 9000. Al final de cada subrutina se retorna al menú y si en éste se responde con 9 acabará el programa.

El programa en Pascal está incluido en el apéndice correspondiente con el nombre AL_PREPARA y, como es habitual, procederemos a comentar las instrucciones principales.

```
RESET(cliente,'cliente.dat');
```

Abrimos el fichero CLIENTES.DAT.

```
SEEK(cliente,1);
GET(cliente);
actual:=cliente^.primero;
ultimo:=cliente^.maximo;
```

Leemos el primer registro para poder almacenar, en la variable ACTUAL, el número del primer registro a grabar y en la variable ULTIMO el número de registros que puede contener el fichero.

```
REPEAT
  menu;
  CASE que_quiere OF
       '1':carga;
```

```
      '2':lectura;
      '3':nombre;
      '4':modificacion;
      '5':borrado;
    END;
```

Cedemos el control al procedimiento MENU y, basándonos en la respuesta dada, llamamos uno de los cinco procedimientos.

```
UNTIL que_quiere='9';
```

A no ser que se responda con 9, en cuyo caso terminará el programa.

Subrutina CARGA

Una vez borrada la pantalla se visualizará el número del primer registro que será objeto de grabación. Ello puede ser interesante cuando al reanudar la carga queramos saber cuántos registros hemos cargado ya y luego valorar cuánto se empleará para cargar el resto. Si se está en una fase de carga avanzada, será útil saber cuál fue el último registro cargado y comprobar si la operación se realizó de forma correcta. Se entra luego en un bucle de instrucciones en donde se solicitan los datos del cliente. El bucle termina cuando se responde con "zzzz" a la petición del nombre del cliente. El control se cede a la subrutina CLAVE que tiene el objeto de encontrar en el fichero CLIENTES el nombre del cliente.

En un fichero de acceso mediante clave es importante que no existan claves idénticas, puesto que no sería posible distinguirlas, y por ello rechazaremos durante la operación de carga las claves que existan ya en el fichero. Si hay espacio en disco, el control se cederá a la subrutina ACT_NUDO, que busca el espacio para incluir el cliente y luego, mediante la subrutina TECLADO,, se introducen los datos del cliente. Después se escribe el registro del nuevo cliente y, finalmente, se actualiza el primer registro del fichero CLIENTES, en donde se almacenó el siguiente registro a grabar.

Las descripción de esta subrutina se hizo ya en el primer volumen y se puso de manifiesto el recurso a la subrutina CLAVE.

Subrutina ACT_NUDO

El objeto de esta subrutina es introducir la clave en orden ascendente en el fichero CLIENTES.DAT. La primera información se

escribe en el registro 1 y los punteros izquierdo y derecho se pondrán a 0. Para las sucesivas informaciones se examinará el árbol desde el principio y se comparará el contenido de la variable NOMINATIVO con el contenido de la variable NOMBRE. Pueden verificarse dos casos: que el valor de la variable NOMINATIVO sea mayor que el de la variable NOMBRE o que el valor de NOMINATIVO sea inferior al de NOMBRE.

En el primer caso (NOMINATIVO>NOMBRE) se pueden producir dos situaciones: si el puntero derecho contiene un valor mayor que 0 se continúa la búsqueda a partir del nudo indicado por el puntero derecho, y si, por el contrario, el puntero derecho contiene un valor negativo o el valor 0 se almacenará el NOMINATIVO en el nudo apuntado por "ACTUAL"; además, se actualizará el nudo apuntado por "P".

En el segundo caso (NOMINATIVO<NOMBRE) se pueden tener también dos situaciones: si el puntero izquierdo contiene un valor diferente de 0 se continuará la búsqueda a partir del nudo indicado por el puntero izquierdo, y si dicho puntero contiene el valor 0 se almacenará el "nominativo" en el nudo apuntado por la variable ACTUAL, mientras que el nudo apuntado por "P" será objeto de actualización. Ha de destacarse que el puntero izquierdo estará puesto a 0 mientras que el puntero derecho contenga un valor negativo.

Esta subrutina está incluida en el programa AL_GESTION que figura en el apéndice y ocupa las instrucciones 1200, a 12540. Pasamos a comentar las instrucciones principales:

```
12040 HECHO=0
12050 IF ACTUAL%=2 THEN LSET SI$=MKI$(0):LSET DE$=
      MKI$(0):GOTO 12520
```

Si se tiene que grabar el primer nudo, la raíz del árbol, se pondrán a 0 los dos punteros y se hará una nueva entrada de la subrutina. De otro modo:

```
12060 P%=2
12070 REM
12080 NUMERO%=P%
12090 GOSUB 15000:REM -->LECTURA CLIENTE
12100 IF NOMINATIVO$=CL$ THEN GOTO 12300
```

Después de haber posicionado de forma adecuada la variable P%, se da comienzo a un bucle de instrucciones para la búsqueda del nudo en donde insertar el nuevo "nominativo". En el caso de que este último sea "mayor" que el nudo actual se cederá

el control a la instrucción 12300. Si el nominativo es "menor" que el almacenado en el nudo se tendrá:

```
12110 IF CVI(SI$)<>0 THEN P%=CVI(SI$):GOTO 12500
```

Si el puntero izquierdo contiene un valor diferente de 0 ponemos en P% el valor del puntero izquierdo y si HECHO sigue puesto a 0 volveremos a recorrer el árbol; de otro modo:

```
12130 LSET SI$=MKI$(ACTUAL%)
12140 GOSUB 15500:REM -->ESCRITURA CLIENTE
```

Actualizamos el nudo apuntado por P%, lo escribimos en disco y luego:

```
12145 NUMERO%=ACTUAL%
12146 GOSUB 15000:REM -->LECTURA CLIENTE
12150 LSET SI$=MKI$(0)
12160 LSET DE$=MKI$(-P%)
12170 GOTO 12500
12500 REM
12510 IF HECHO=0 THEN GOTO 12080
12520 REM
12530 RETURN
```

Después de haberlo leído a partir del disco actualizamos el nudo apuntado por ACTUAL% (destaquemos la forma en que se carga el puntero derecho). A continuación, se realiza una nueva entrada de la subrutina.

Nos queda todavía por examinar el caso en el que el "nominativo" tenga un valor mayor que el nudo recorrido, y para ello hay que examinar las instrucciones 12300/12500 del apéndice.

```
12310 IF CVI(DE$)>0 THEN P%=CVI(DE$):GOTO 12500
```

Si el puntero derecho contiene un valor mayor que 0, ponemos en P% el puntero derecho y pasamos al nudo correspondiente volviendo a entrar en el ciclo, o de otro modo:

```
12330 M%=CVI(DE$)
12340 LSET DE$=MKI$(ACTUAL%)
12350 GOSUB 15500:REM -->ESCRITURA CLIENTE
```


Actualizamos y escribimos en disco el nudo apuntado por P%, no antes de haber almacenado en M% el puntero derecho de dicho nudo, y luego tendremos:

```
12356 GOSUB 15000:REM -->LECTURA CLIENTE
12360 LSET DE$=MKI$(M%)
```

Actualizamos el nudo apuntado por ACTUAL%, después de haberlo leído a partir del disco.

El procedimiento ACT_NUDO está incluido en el apéndice destinado al lenguaje Pascal y, como es habitual, procedemos a comentar las instrucciones principales:

```
hecho:=FALSE;
IF actual=2
THEN BEGIN  {si primer registro}
 izquierdo:=0;
 derecho:=0;
END
```

En el caso que se tenga que grabar el primer nudo, la raíz del árbol, se pondrán a 0 los dos punteros y se realizará una nueva entrada de la subrutina, o, de otro modo:

```
ELSE BEGIN
   p:=2;
   REPEAT
      SEEK(cliente,p);
      GET(cliente);
      IF nominativo>cliente
      THEN BEGIN
```

Una vez que se haya cargado en la variable "P" el valor 2 se da comienzo a un bucle de instrucciones para la búsqueda del nudo en donde introducir el nuevo "nominativo". En el caso de que este último sea "mayor" que el nudo actual, se ejecutan las instrucciones siguientes:

```
IF derecho>0
  THEN p:=derecho
```

Si el puntero derecho contiene un valor mayor que 0 ponemos en "P" este valor y leemos el nudo, volviendo a entrar en el ciclo. De otro modo:

```
ELSE BEGIN
    hecho:=TRUE;
    m:=derecho;
    derecho:=actual;
    SEEK(cliente,p);
    PUT(cliente);
```

Actualizamos y escribimos en disco el nudo apuntado por "P", no antes de haber almacenado en "M" el puntero derecho de dicho nudo; luego tendremos:

```
        derecho:=m;
        izquierdo:=0;
    END;
END
```

Actualizamos el nudo apuntado por ACTUAL. Saldremos del bucle solamente si la variable booleana HECHO está puesta a 1, es decir, en el caso de que el nudo haya sido objeto de actualización.

Nos queda todavía por examinar el caso en el que el "nominativo" tenga un valor menor que el nudo recorrido:

```
ELSE BEGIN
IF izquierdo<>0
  THEN p:=izquierdo;
```

Si el puntero izquierdo contiene un valor diferente de 0 ponemos en "P" este valor y volvemos a recorrer el árbol. De otro modo:

```
ELSE BEGIN
    hecho:=TRUE;
    izquierdo:=actual;
    SEEK(cliente,p);
    PUT(cliente);
```

Actualizamos el nudo apuntado por "P" y lo escribimos en disco. Tendremos:

```
        izquierdo:=0;
        derecho:=-p;
    END;
END;
```


Actualizamos el nudo apuntado por "P". Luego se sale de la subrutina.

Subrutina CLAVE

Esta subrutina tiene como objeto buscar en el fichero CLIENTES una clave que corresponda al contenido de la variable NOMINATIVO. El árbol binario se recorre desde el principio y se busca el NOMINATIVO con el empleo de la técnica que veremos a continuación.

La búsqueda se detiene cuando se encuentra el "nominativo" o cuando se llega al final del árbol (J=0). La variable FIN nos indica si no se encontró el "nominativo". La variable PROXIMO indica el último nudo que se recorrió y por ello es de gran utilidad en la búsqueda de claves parciales.

Esta subrutina está incluida en el apéndice destinado al BASIC en el programa AL_GESTION, en las líneas desde 13000 a 13330. Pasamos a comentar las instrucciones principales.

```
13040 ENC=0
13050 FIN=0
13060 ESTO%=2
13070 NUMERO%=2
13075 IF LEN(NOMINATIVO$))=31 THEN GOTO 13078
13077 NOMINATIVO$=NOMINATIVO$+" ":GOTO 13075
13078 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,4 AS
      TE$,4 AS PR$,1 AS TI$,2 AS SI$,2 AS DE$
13080 IF NUMERO%=0 THEN GOTO 13200
```

Después de las siempre necesarias inicializaciones de algunas variables se observa si la variable NUMERO% contiene el valor 0, en cuyo caso se realiza una nueva entrada de la subrutina, recordando en la variable PROXIMO% el número que sigue. En otro caso:

```
13090 GOSUB 15000:REM -->LECTURA CLIENTE
13100 ESTO%=NUMERO%
13110 IF NOMINATIVO$<CL$ THEN NUMERO%=CVI(SI$):
      GOTO 13080
```

Se lee un registro de CLIENTE y si el valor contenido en la variable NOMINATIVO$ es inferior al almacenado en el nudo se transfiere el puntero izquierdo del nudo a la variable NUMERO% y se cede el control a la instrucción 13080. Si no es así:

```
13120 IF NOMINATIVO$=CL$ THEN ENC=1:PROXIMO%=CVI(
      DE$):GOTO 13300
```

Si se encuentra el "nominativo" se pondrá a 1 la variable ENC, se actualizará la variable PROXIMO% de modo que indique el nudo que sigue y se realizará una nueva entrada de la subrutina. De otra forma::

```
13130 IF CVI(DE$)>0 THEN NUMERO%=CVI(DE$):GOTO 13080
```

Si el valor contenido en la variable NOMINATIVO$ es inferior al valor contenido en el nudo y, al mismo tiempo, el puntero derecho del nudo indica que siguen otros nudos, actualizaremos la variable NUMERO% y repetiremos el procedimiento antes expuesto. De no ser así:

```
13140 NUMERO%=0
13200 FIN=1
13210 PROXIMO%=ESTO%
13220 RETURN
```

No fue posible encontrar el "nominativo" y, por ello, se indica tal circunstancia en FIN, acabando con una nueva entrada de la subrutina.

El procedimiento CLAVE está incluido en el apéndice destinado al lenguaje Pascal en el programa AL_GESTION. Como es habitual, procederemos a comentar las instrucciones principales.

```
enc:=FALSE;
fin:=FALSE;
esto:=2;
j:=2;
WITH cliente^
DO BEGIN
 REPEAT
  IF j=0
  THEN BEGIN
    fin:=TRUE;
    proximo:=esto;
  END;
```

Después de las inevitables inicializaciones de las variables, se comprueba si la variable "J" contiene el valor 0, en cuyo caso se realiza el salto a la subrutina recordando en la variable PROXIMO el nudo que sigue. De no ser así:

```
ELSE BEGIN
SEEK(cliente,j);
GET(cliente);
esto:=j;
IF nominativo<cliente
 THEN j:=izquierdo
```

Se lee un registro de CLIENTE. Si la variable NOMINATIVO contiene un valor menor que el del registro leído se prosigue con la búsqueda en el siguiente nudo extrayendo el valor del puntero IZQUIERDO. En caso contrario:

```
ELSE IF nominativo=cliente
     THEN BEGIN
         enc:=TRUE;
         proximo:=derecho;
       END
```

Si se encuentra el "nominativo" se pondrá en "verdadero" la variable booleana ENC y se graba en PROXIMO el nudo que sigue; finalmente, se produce el reenvío a la subrutina. De no ser así:

```
  ELSE IF derecho>0 THEN j:=derecho
                    ELSE j:=0;
END;
```

Si el "nominativo" a buscar es mayor que el existente en el nudo, examinaremos el siguiente nudo solamente si el puntero derecho contiene un valor mayor que 0; en otro caso, se saldrá de la subrutina. Se sale del bucle de instrucciones en caso de que las variables ENC o FIN estén puestas a 1. Si se encuentra el "nominativo" se leerá el registro para ponerlo a disposición del usuario.

Subrutina LECTURA

Permite leer un determinado número de registros y visualizarlos en la pantalla. Se solicita el número del primero y del último registro a leer.

El procedimiento LISTA tiene como objeto encontrar el enésimo registro, a condición de que esté grabado. Si no existe el re-

gistro se activará la variable FIN. Se sale de la subrutina cuando se pulsa la tecla <Escape>, cuando se ha visualizado el último registro solicitado o cuando se han visualizado todos los registros existentes en el fichero.

Esta subrutina está incluida en el programa AL_GESTION y se comentó a su debido tiempo.

Subrutina LISTA

Sirve para leer el enésimo registro a condición de que esté grabado. Si el registro no se grabó se activará la variable FIN.

Subrutina NOMBRE

Permite leer los clientes mediante el campo clave, o sea, con el nombre. La subrutina, por medio de TECLADO, pregunta desde qué nombre hasta qué otro se quiere leer. A continuación, el control se cede a la subrutina CLAVE, que se encargará de encontrar el primer "nominativo". En el caso de búsqueda con claves parciales (por ejemplo, si queremos partir de las claves que comienzan con "b"), al no encontrar la clave correspondiente se activará la variable FIN y se cederá el control a la subrutina BUSQUEDA, que se posiciona en el último registro encontrado.

Se entra luego en un bucle en el que se visualiza el registro CLIENTE y se cede el control a la subrutina BUSQUEDA, que encuentra el registro siguiente. Se sale de la subrutina cuando se pulsa la tecla <Escape>, cuando se visualizó el último "nominativo" o cuando se visualizaron todos los registros existentes en el fichero.

La subrutina está incluida desde la instrucción 6000 a la instrucción 6540 del programa AL_GESTION incluido en el apéndice reservado al lenguaje BASIC. Veamos las instrucciones principales:

```
6090 PRINT "NOMBRE DEL PRIMER CLIENTE :"
6100 SW=3
6110 XPOS=32
6120 YPOS=20
6130 GOSUB 1000:REM --->TECLADO
6140 PARTIDA$=ALFA$
```

Mediante la subrutina TECLADO se solicita el nombre del primer cliente para almacenarlo en la variable PARTIDA$. De forma

análoga, se solicita el nombre del último cliente a visualizar y se almacena en la variable TERMINO$

```
6235 NOMINATIVO$=PARTIDO$
6240 GOSUB 13000:REM -->CLAVE
6250 IF FIN=1 THEN GOTO 6500
```

Con la ayuda de la subrutina CLAVE buscamos el "nominativo" y, si no lo encontramos, cedemos el control a la subrutina BUSQUEDA, que nos permite encontrar el nudo anterior. Si se encuentra el "nominativo" se tendrá:

```
6260 GOSUB 500:REM ---->BORRADO PANTALLA
6270 GOSUB 10000:REM -->MASCARA
6280 GOSUB 11000:REM -->VISUALIZACION
```

Se visualiza el registro en la pantalla.

```
6287 GOSUB 13500:REM -->BUSQUEDA
6290 YPOS=20:XPOS=1
6300 GOSUB 15800:REM -->POSICIONA EL CURSOR
6310 PRINT "<ESPACIO> PARA CONTINUAR,<ESC>
     PARA TERMINAR:";
6320 GOSUB 15900:REM -->LECTURA CARACTER
```

Se lee el siguiente registro y luego se da la posibilidad al usuario de observar con comodidad la pantalla y decidir si se ven los registros siguientes o se realiza el reenvío de la subrutina.

El procedimiento NOMBRE está incluido en el programa AL_GESTION del apéndice reservado al lenguaje Pascal. Veamos las instrucciones principales:

```
GOTOXY(0,20);
WRITE('nombre del primer cliente  :');
sw:='3';
x_pos:=32;
y_pos:=20;
teclado;
partida:=alfa;
```

Mediante el procedimiento TECLADO se pide el nombre del primer cliente para almacenarlo en la variable PARTIDA. De forma análoga se solicita el nombre del último cliente a visualizar y se almacena en la variable TERMINO.

```
nominativo:=partida;
clave;
IF fin
THEN BEGIN
 busqueda;
 fin:=FALSE;
END;
```

Con la ayuda del procedimiento CLAVE buscamos el "nominativo" y, si no lo encontramos, cedemos el control al procedimiento BUSQUEDA, que procede a encontrar el nudo anterior. Si lo encuentra tendremos:

```
REPEAT
 borrado_pantalla;
 mascara;
 visualizacion;
```

Se visualiza el registro en la pantalla.

```
esto:=proximo;
IF esto<>0 THEN busqueda;
WRITE('<ESPACIO> para continuar,',
      '<ESC> para terminar');
READ(respuesta);
```

Se lee el siguiente registro y luego se da la posibilidad al usuario de observar con comodidad la pantalla para decidir si examina los siguientes registros o si efectúa el reenvío de la subrutina.

```
UNTIL(respuesta=CHR$(27))
   OR(esto=0)
   OR(cliente^.cliente>termino)
```

El bucle termina con la indicación por parte del usuario (cuando pulsa la tecla <Escape>), cuando se visualiza el último cliente solicitado o, finalmente, cuando se acaban los registros en el fichero CLIENTES.

Subrutina BUSQUEDA

Sirve para leer el registro del fichero CLIENTES que corresponde al contenido de la variable PROXIMO. En el caso de que el puntero izquierdo sea diferente de 0, se continuará la búsqueda basándose en su contenido. La variable PROXIMO al final de la subrutina se actualiza de modo que apunte el registro siguiente. La subrutina BUSQUEDA, incluida en el apéndice reservado al BASIC, se introduce en el programa AL_GESTION ocupando las líneas 13500/13820.

```
13540 FUERA=0
13550 REM
13560 ESTO%=ABS(PROXIMO%)
13565 NUMERO%=ESTO%
13570 GOSUB 15000:REM -->LECTURA CLIENTE
```

Una vez inicializada la variable FUERA se lee un registro del fichero CLIENTES. El número del registro leído depende del valor absoluto contenido en la variable PROXIMO%, puesto que puede contener valores negativos.

```
13580 IF CVI(SI$)<>0 AND PROXIMO%>=0 THEN PROXIMO%=
      CVI(SI$):GOTO 13800
```

Se almacena el siguiente nudo con tal de que el puntero izquierdo no esté puesto a 0 y la variable PROXIMO% no contenga un valor negativo y luego se vuelve a realizar la lectura. De no ser así:

```
13590 FUERA=1
13600 PROXIMO%=CVI(DE$)
```

Se almacena el siguiente nudo y se hace una nueva entrada en la subrutina.

El procedimiento BUSQUEDA se incluyó en el programa AL_GESTION del apéndice reservado al lenguaje Pascal. Veamos las instrucciones principales:

```
fuera:=FALSE;
REPEAT
  esto:=ABS(proximo);
  SEEK(cliente,esto);
  GET(cliente);
```

Después de inicializar la variable FUERA se comienza un bucle de instrucciones en donde se lee un registro de CLIENTE. El número del registro depende del valor absoluto de la variable PROXIMO, habida cuenta de que puede contener valores negativos.

```
IF (izquierdo<>0) AND (proximo>=0)
  THEN proximo:=izquierdo;
```

Se lee el siguiente nudo a condición de que el puntero izquierdo contenga un valor diferente de 0 y de que la variable PROXIMO contenga un valor mayor o igual que 0.

```
      ELSE BEGIN
      fuera:=TRUE;
      proximo:=derecho;
      END;
  UNTIL fuera;
END;{with cliente^}
```

Se busca el siguiente nudo y se sale del procedimiento. Hay que destacar el hecho de que se sale del procedimiento solamente cuando la variable booleana FUERA está en VERDADERO.

Subrutina MODIFICACION

El acceso mediante árbol binario nos permite modificar cualquier registro. Después de borrar la pantalla se solicita mediante la subrutina TECLADO el nombre del cliente a modificar. La subrutina CLAVE se encarga de buscar dicho "nominativo" en el fichero CLIENTES. Si no la encuentra dará la indicación correspondiente en la pantalla y saldrá de la subrutina. Si se encontrara será posible modificar todos los campos a excepción del correspondiente al nombre del cliente. El cursor se posiciona al comienzo del campo si el operador pulsa la tecla <Escape> y podrá modificarlo (escribir los datos nuevos y terminar con <Return>). Finalmente, el registro se vuelve a escribir en disco exactamente en la misma posición.

Esta subrutina fue objeto de comentario en el primer volumen.

Subrutina de borrado

Podemos indicar los registros que han de considerarse borrados. Hemos dicho ya que para no complicar los programas no empleamos la técnica de recuperación inmediata del espacio libre dejado por los borrados; será precisamente el programa AL_COMPACTA el instrumento para la recuperación de dicho espacio.

Después del borrado de la pantalla se solicitará la clave de BUSQUEDA mediante la subrutina TECLADO. El control se cederá a la subrutina CLAVE. Si no se encontrara el nombre se dará la indicación correspondiente en la pantalla y se saldrá de la subrutina. Si se encuentra la clave se visualiza el registro en la pantalla. A continuación se pide la confirmación de borrado al usuario. Si este último responde con "S" o con "s" el campo TIPO se modificará en "C" y luego el registro se grabará en disco en la misma posición.

También esta subrutina se comentó con anterioridad.

Programa AL_COMPACTA

Antes de ejecutar el programa es preciso realizar las operaciones preliminares siguientes: cambiar el nombre del fichero CLIENTES.DAT por CLIENTES.VIE y llamar el programa AL_PREPARA, que establece el nuevo fichero CLIENTES.DAT. En BASIC haremos RENAME CLIENTES.DAT CLIENTES.VIE. En Pascal, utilizaremos la función C)ambio del F)ichero y luego escribiremos CLIENTES.DAT, CLIENTES.VIE. Luego se llama el programa AL_PREPARA para preestablecer el fichero CLIENTES.DAT nuevo. Durante esta fase es posible dimensionar el fichero de manera diferente.

El programa está incluido en el apéndice destinado al lenguaje BASIC con el nombre de AL_COMPACTA. Procedamos a comentar las instrucciones principales:

```
140 OPEN "R",2,"CLIENTES.DAT",92
160 OPEN "R",1,"CLIENTE.VEC",92
```

Se abren los dos ficheros declarándolos de tipo aleatorio.

```
175 GET #2,1
180 FIELD 1,2 AS ACT$,2 AS ULT$
190 GET #1,1
210 X%=CVI(ULTIMO$)
220 Y%=CVI(ACT$)
```

Se lee el primer registro del fichero VIEJOS para conocer el número de los registros allí grabados. Del nuevo fichero CLIENTES se toma el número de registro que puede contener y dicho valor se almacena en la variable X%. Luego, después de haber inicializado las variables NUMERO%, ACTUAL% y Z%, iniciamos un bucle de instrucciones que termina después de haber leído el último registro del fichero VIEJOS.

```
280 GET #1,Z%
290 IF TIP$<>"C" THEN GOTO 320
300 PRINT CLI$;" BORRADO"
```

Se lee un registro del fichero VIEJOS y se lleva la indicación de borrado a la pantalla, incrementándose la variable Z% y cediendo el control a la instrucción 480. De otro modo:

```
330 IF Z%>X% THEN GOTO 500
```

Si se han superado las dimensiones del fichero se da la indicación correspondiente y se cierra el programa. De no ser así:

```
350 GOSUB 12000:REM -->ACT NUDO
```

El control del programa se cede a la subrutina ACT_NUDO, que se encarga de encontrar el lugar en donde introducir el cliente.

```
360 LSET CL$=CLI$
```

Al primer campo del registro CLIENTES se transfiere el contenido del primer campo del registro VIEJOS. A continuación se dan otras instrucciones, similares a ésta, para completar el registro.

```
450 PRINT CL$
460 NUMERO%=ACTUAL%
465 GOSUB 15500:REM -->ESCRITURA CLIENTE
470 ACTUAL%=ACTUAL%+1
```

Se imprime el nombre del cliente, se graba el fichero CLIENTES y se actualizan los contadores de registros.

```
480 IF Z%<Y% THEN GOTO 260
490 GOTO 600
```


Si hay otros registros se cede el control a la instrucción 260, y de no ser así, se tendrá:

```
620 NUMERO%=1
625 GOSUB 15000:REM -->LECTURA CLIENTE
630 LSET ACT$=MKI$(ACTUAL%)
640 GOSUB 15500:REM -->ESCRITURA CLIENTE
```

Grabamos la situación de los registros válidos en el primer registro del fichero CLIENTES.

```
650 CLOSE 1
660 CLOSE 2
670 END
```

Cerramos los dos ficheros.

La subrutina ACT_NUDO se comentó ya en la explicación del programa AL_GESTION.

En el apéndice destinado al lenguaje Pascal hemos incluido el programa AL_COMPACTA. Veamos las instrucciones principales:

```
RESET(viejo,'cliente.vie');
RESET(cliente,'cliente.dat');
```

Se abren los dos ficheros.

```
SEEK(cliente,1);
GET(cliente);
SEEK(viejo,1);
GET(viejo);
x:=cliente^.maximo;
y:=viejo^.primero;
```

Se lee el primer registro del fichero CLIENTES para conocer el número de los registros allí grabados. Se almacena en la variable "X" el número máximo de registros, leído desde el primer registro del fichero CLIENTES. Luego, después de haber inicializado las variables ACTUAL e "Y", se da comienzo a una serie de instrucciones que terminan cuando se ha leído el último registro del fichero VIEJOS.

```
GET(viejo);
actual:=2;
```

```
z:=2;
REPEAT
 WITH viejo^
 DO BEGIN
       IF tipo='C'
       THEN WRITELN(cliente:15,' borrado')
```

Se lee un registro del fichero VIEJOS y si este último lleva la indicación de borrado, se dará la señal adecuada en la pantalla. De no ser así se tendrá:

```
ELSE BEGIN
IF actual-1 > x
THEN BEGIN
  WRITELN('hay ',actual-1,' registros',
          'activos enun fichero que los',
          'prevee ',x,'!');
  WRITELN('no puedo proseguir');
  WRITELN('pulse <RETURN>');
  READLN;
  EXIT(program);
END;
```

Si no hay suficiente espacio en disco se emitirá la indicación correspondiente y se bloqueará el programa. De otro modo:

```
ELSE BEGIN
 nominativo:=cliente;
 act(nudo;
```

Se cede el control al procedimiento ACT_NUDO que se comentó con anterioridad. Siguen otras instrucciones para transferir cada campo desde un registro a otro.

```
  WRITELN(cliente:15);
  SEEK(cliente,actual-1);
  PUT(cliente);
  actual=actual+1
END;
```

En la pantalla se escribe el nombre del cliente, se graba el fichero CLIENTES y se actualiza el contador de registros.


```
      END;
      z:=z+1:
      GET(viejo);
    END;  {de la with viejo^}
  UNTIL z>=y;
```

Si hay otros registros en el fichero se procesan. En otro caso:

```
SEEK(cliente,1);
GET(cliente);
cliente^.primero:=actual;
SEEK(cliente,1);
PUT(cliente);
```

Grabamos en el primer registro del fichero CLIENTES cuál será el siguiente registro a grabar.

```
CLOSE(cliente,LOCK);
CLOSE(viejo);
```

Cerramos los dos ficheros.

Después de las oportunas comprobaciones será posible borrar el fichero CLIENTES.VIE. En lenguaje BASIC haremos ERASE CLIENTES.VIE y en Pascal llamaremos a la función R)etirada del F)ichero, escribiremos CLIENTES.VIE y responderemos con "S" a la solicitud de confirmación del borrado.

APENDICE A

APENDICE BASIC

IS_PREPARA

```
100 REM ************************
110 REM **      IS_PREPARA    **
120 REM ************************
130 REM
140 FILE$="CLIENTES.DAT"
160 OPEN "R",1,FILE$,88
170 INPUT "¿DE CUANTOS REGISTROS ESTARA
    COMPUESTO EL FICHERO?";ULTIMO
180 PRIMERO=2
190 FIELD 1,2 AS PRIMERO$,2 AS ULTIMO$
200 LSET PRIMERO$=MKI$(PRIMERO)
210 LSET ULTIMO$=MKI$(ULTIMO)
220 PUT #1,1
230 FIELD 1,31 AS CLIENTE$,33 AS DIRECCION$,
    15 AS CIUDAD$, 4 AS TELEFONO$,4 AS PREFIJO$,
    1 AS TIPO$
250 LSET CLIENTE$=""
260 LSET DIRECCION$=""
270 LSET CIUDAD$=""
280 LSET TELEFONO$="0"
280 LSET PREFIJO$="0000"
300 LSET TIPO$=" "
320 FOR X%=2 TO ULTIMO+1
330 PUT #1,X%
```

```
340 NEXT X%
350 CLOSE #1
400 OPEN "R",2,"INDICE.DAT",35
410 FIELD 2,31 AS NOMBRE$,2 AS IZQUIERDO$,2 AS DERECHO$
420 LSET NOMBRE$=""
430 LSET IZQUIERDO$=MKI$(0)
440 LSET DERECHO$=MKI$(0)
450 FOR X%=I TO ULTIMO
460 PUT #2,X%
470 NEXT X%
480 CLOSE #2
500 END
```

IS_GESTION

```
100 REM ********************
110 REM **   IS_GESTION   **
120 REM ********************
130 REM
140 OPEN "R",1,"CLIENTES.DAT",88
170 NUMERO%=1
180 FIELD 1,2 AS ACTUAL$,2 AS ULTIMO$
190 GOSUB 15000:REM  -->LECTURA CLIENTE/1
200 ACTUAL%=CVI(ACTUAL$)
210 ULTIMO%=CVI(ULTIMO$)
211 OPEN "R",2,"INDICE.DAT",25
212 FIELD 2,31 AS NO$,2 AS SI$,2 AS DE$
220 GOSUB 500:REM ----->BORRADO PANTALLA
230 GOSUB 800:REM ----->MENU
240 ON QUEQUIERE GOSUB 2000,4000,6000,9000
250 IF QUEQUIERE<>9 THEN GOTO 220
270 CLOSE 1
280 CLOSE 2
290 END
500 REM :::::::::::::::::::::::
510 REM ::  BORRADO PANTALLA  ::
520 REM :::::::::::::::::::::::
530 REM
540 CLS
550 RETURN
```



```
800 REM ::::::::::::
810 REM ::  MENU  ::
820 REM ::::::::::::
830 REM
840 PRINT "1 --> CARGA DE DATOS"
850 PRINT
860 PRINT "2 --> LECTURA CON EL NUMERO DE REGISTRO"
870 PRINT
880 PRINT "3 --> LECTURA CON EL NUMERO DE CLIENTE"
890 PRINT
900 PRINT "4 --> MODIFICACION DE UN REGISTRO"
910 PRINT
920 PRINT "5 --> BORRADO DE UN REGISTRO"
930 PRINT
940 PRINT "9 --> FIN DEL PROGRAMA"
950 PRINT
960 PRINT "?":RESPUESTA$=INPUT$(1):QUEQUIERE=
    ASC(RESPUESTA$)-ASC("0")
980 RETURN
1000 REM ::::::::::::::
1010 REM ::  TECLADO  ::
1020 REM ::::::::::::::
1030 REM
1040 ALFA$=""
1050 GOSUB 15800:REM -->POSICIONA EL CURSOR
1060 GOSUB 15900:REM -->LECTURA CARACTER
1070 IF ASC(CAR$)=13 THEN GOTO 1210
1080 IF ASC(CAR$)<>8 THEN GOTO 1310
1090 REM -----------------------
1100 REM --  TECLA DE RETORNO  --
1110 REM -----------------------
1120 IF LEN(ALFA$)>1 THEN ALFA$=LEFT$(ALFA$,
     LEN(ALFA$)-1) ELSE ALFA$=""
1130 REM --------------------------
1140 REM --  IMPRESION DEL CAMPO  --
1150 REM --------------------------
1160 GOSUB 15800:REM -->POSICIONA EL CURSOR
1170 PRINT ALFA$;' ';
1180 GOSUB 15800:REM -->POSICIONA EL CURSOR
1190 PRINT ALFA$;
1200 GOTO 1060
```

```
1210 REM ---------------------
1220 REM --  NUEVA ENTRADA  --
1230 REM ---------------------
1240 IF SW=3 THEN RETURN
1250 G#=0
1260 FOR X=1 TO LEN(ALFA$)
1270 G#=G#*10+ASC(MID$(ALFA$,X,1))-48
1280 NEXT X
1290 IF SW=1 THEN PEQUENO=CSNG(G#) ELSE GRANDE#=G#
1300 RETURN
1310 REM -----------------------
1320 REM --  CARACTER NORMAL  --
1330 REM -----------------------
1340 IF SW=3 THEN GOTO 1450
1350 REM -------------------------
1360 REM --  CARACTER NUMERICO  --
1370 REM -------------------------
1380 IF CAR$<"0" OR CAR$>"9" THEN GOTO 1400
1390 GOTO 1490
1400 REM ----------------------
1410 REM --  ERROR DE TECLA  --
1420 REM ----------------------
1430 PRINT CHR$(7);
1440 GOTO 1060
1450 REM ---------------------------
1460 REM --  CARACTER ALFABETICO  --
1470 REM ---------------------------
1480 IF CAR$<" " OR CAR$>"z" THEN GOTO 1400
1490 ALFA$=ALFA$+CAR$
1500 GOTO 1130
2000 REM :::::::::::::::::::::
2010 REM ::  CARGA DE DATOS  ::
2020 REM :::::::::::::::::::::
2030 REM
2040 GOSUB 500:REM ----->BORRADO PANTALLA
2050 PRINT "LA CARGA PARTE CON EL NUMERO:";ACTUAL%-1
2060 IF ACTUAL%<3 THEN  GOTO 2140
2070 NUMERO%=ACTUAL%-1
2080 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,4 AS TE$
     4 AS PR$,1 AS TI$
2090 GOSUB 15000:REM -->LECTURA CLIENTE
2100 PRINT "EL REGISTRO ANTERIOR ESCRITO
     ES EL SIGUIENTE:"
2110 GOSUB 10000:REM -->MASCARA
2120 GOSUB 11000:REM -->VISUALIZACION
2130 PRINT
2140 INPUT "PULSE <RETURN> PARA INICIAR LA CARGA";
     RESPUESTA$
2150 REM
2155 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,4 AS TE$
     4 AS PR$,1 AS TI$
2160 GOSUB 500:REM ---->BORRADO PANTALLA
2170 YPOS=20:XPOS=1
2180 GOSUB 15800:REM -->POSICIONA EL CURSOR
2190 PRINT "PARA TERMINAR,A LA PETICION DE CLIENTE
     RESPONDA 'zzzz'"
2200 GOSUB 10000:REM -->MASCARA
2210 XPOS=13:YPOS=4
2220 SW=3
2230 GOSUB 1000:REM --->TECLADO
2240 NOMINATIVO$=ALFA$
2250 IF NOMINATIVO$="zzzz" THEN RETURN
2260 IF ACTUAL%>ULTIMO% THEN INPUT"NO PUEDO SEGUIR
     PUES SE HA SUPERADO LA DIMENSION MAXIMA DEL
     FICHERO.PULSE <RETURN>";RESPUESTA$:RETURN
2280 GOSUB 13000:REM -->CLAVE
2290 IF ENC=0 THEN GOTO 2310
2295 YPOS=20:XPOS=1
2296 GOSUB 15800:REM -->POSICIONA EL CURSOR
2300 INPUT "ESTE CLIENTE YA EXISTE.NO PUEDO
     INTRODUCIRLO.PULSE <RETURN>           ";
     RESPUESTA$
2305 RETURN
2310 REM
2320 CLIENTE$=NOMINATIVO$
2330 SW=3
2340 XPOS=13
2350 YPOS=6
2360 GOSUB 1000:REM --->TECLADO
2370 DIRECCION$=ALFA$
2380 SW=3
2390 XPOS=13
```

```
2400 GOSUB 1000:REM --->TECLADO
2420 CIUDAD$=ALFA$
2430 SW=2
2440 XPOS=13
2450 YPOS=10
2460 GOSUB 1000:REM --->TECLADO
2470 TELEFONO#=GRANDE#
2480 SW=3
2490 XPOS=13
2400 YPOS=12
2510 GOSUB 1000:REM --->TECLADO
2520 PREFIJO$=ALFA$
2530 TIPO$=" "
2590 LSET CL$=CLIENTE$
2600 LSET IN$=DIRECCION$
2610 LSET CI$=CIUDAD$
2620 LSET TE$=MKI(TELEFONO#)
2630 LSET PR$=PREFIJO$
2640 LSET TI$=TIPO$
2650 NUMERO%=ACTUAL%
2670 GOSUB 15500:REM -->ESCRITURA CLIENTE
2680 REM
2690 FIELD 1,2 AS ACTUAL$,2 AS ULTIMO$
2700 LSET ACTUAL$=MKI$(ACTUAL%)
2710 LSET ULTIMO$=MKI$(ULTIMO%)
2720 NUMERO%=1
2730 GOSUB 15500:REM -->ESCRITURA CLIENTE/1
2735 GOSUB 12000:REM -->ACT_NUDO
2740 GOTO 2150
2750 REM
4000 REM ::::::::::::::::::::::::::::::::::::::::
4010 REM ::  LECTURA CON EL NUMERO DE REGISTRO  ::
4020 REM ::::::::::::::::::::::::::::::::::::::::
4030 REM
4040 YPOS=20:XPOS=1
4050 GOSUB 15800:REM -->POSICIONA EL CURSOR
4060 PRINT "NUMERO DEL PRIMER REGISTRO:"
4070 SW=1
4080 XPOS=32
4090 YPOS=20
4100 GOSUB 1000:REM --->TECLADO
```



```
4110 INICIO%=PEQUENO%+1
4120 YPOS=20:XPOS=1
4130 GOSUB 15800:REM -->POSICIONA EL CURSOR
4140 PRINT "NUMERO DEL ULTIMO REGISTRO:     "
4150 SW=1
4160 XPOS=32
4170 YPOS=20
4180 GOSUB 1000:REM --->TECLADO
4190 ALT%=PEQUENO%+1
4200 IF ALT%>ACTUAL%-1 THEN ALT%=ACTUAL%-1
4210 GOSUB 11500:REM -->LISTA
4220 IF FIN=1 THEN RETURN
4230 REM
4240 GOSUB 500:REM ---->BORRADO PANTALLA
4250 GOSUB 10000:REM -->MASCARA
4260 GOSUB 11000:REM -->VISUALIZACION
4270 YPOS=20:XPOS=1
4280 GOSUB 15800:REM -->POSICIONA EL CURSOR
4290 GOSUB 11500:REM -->LISTA
4300 PRINT "<ESPACIO> PARA SEGUIR,<ESC> PARA
     TERMINAR"
4310 RESPUESTA$=INPUT$(1)
4320 IF ASC(RESPUESTA$)=27 OR FIN=1 OR INICIO%>
     ALT%+1 THEN RETURN
4330 GOTO 4230
6000 REM :::::::::::::
6010 REM ::  NOMBRE  ::
6020 REM :::::::::::::
6030 REM
6060 YPOS=20
6070 XPOS=1
6080 GOSUB 15800:REM -->POSICIONA EL CURSOR
6090 PRINT "NOMBRE DEL PRIMER CLIENTE:"
6100 SW=3
6110 XPOS=32
6120 YPOS=20
6130 GOSUB 1000:REM --->TECLADO
6140 PARTIDA$=ALFA$
6150 YPOS=20
6160 XPOS=1
6170 GOSUB 15800:REM -->POSICIONA EL CURSOR
```

```
6180 PRINT "NOMBRE DEL ULTIMO CLIENTE:
     "
6190 SW=3
6200 XPOS=32
6210 YPOS=20
6220 GOSUB 1000:REM --->TECLADO
6230 TERMINO$=ALFA$
6231 IF LEN(TERMINO$)>=31 THEN GOTO 6235
6232 TERMINO$=TERMINO$+" ":GOTO 6231
6235 NOMINATIVO$=PARTIDA$
6240 GOSUB 13000:REM -->CLAVE
6250 IF FIN=1 THEN GOTO 6500
6260 GOSUB 500:REM ---->BORRADO PANTALLA
6270 GOSUB 10000:REM -->MASCARA
6280 GOSUB 11000:REM -->VISUALIZACION
6282 ESTO%=PROXIMO%
6285 IF ESTO%=0 THEN GOTO 6290
6287 GOSUB 13500:REM -->CERCA
6290 YPOS=20:XPOS=1
6300 GOSUB 15800:REM -->POSICIONA EL CURSOR
6310 PRINT "<ESPACIO> PARA SEGUIR,<ESC>
     PARA TERMINAR:"
6320 GOSUB 15900:REM -->LECTURA CARACTER
6330 IF ASC(CAR$)=27 THEN GOTO 6380
6340 IF CL$>TERMINO$ THEN GOTO 6380
6350 IF ESTO%=0 THEN GOTO 6380
6370 GOTO 6260
6380 REM
6400 RETURN
6410 REM
6500 REM
6510 GOSUB 13500:REM -->BUSQUEDA
6520 FIN=0
6530 GOTO 6260
6540 REM
8000 REM ::::::::::::::::::
8010 REM ::  MODIFICACION  ::
8020 REM ::::::::::::::::::
8030 REM
8040 GOSUB 500:REM ---->BORRADO PANTALLA
8050 YPOS=20:XPOS=1
```



```
8060 GOSUB 15800:REM -->POSICIONA EL CURSOR
8070 PRINT "CLAVE A BUSCAR:"
8080 SW=3
8090 XPOS=23:YPOS=20
8100 GOSUB 1000:REM --->TECLADO
8110 NOMINATIVO$=ALFA$
8130 IF LEN(NOMINATIVO$)=31 THEN GOTO 8150
8140 NOMINATIVO$=NOMINATIVO$+" ":GOTO 8130
8150 GOSUB 13000:REM -->CLAVE
8160 IF FIN=0 THEN GOTO 8210
8170 YPOS=20:XPOS=1
8180 GOSUB 15800:REM -->POSICIONA EL CURSOR
8190 INPUT "NO EXISTE ESTE CLIENTE.PULSE<RETURN>";
     RESPUESTA$
8200 RETURN
8210 REM
8220 GOSUB 10000:REM -->MASCARA
8230 GOSUB 11000:REM -->VISUALIZACION
8240 YPOS=20:XPOS=1
8250 GOSUB 15800:REM -->POSICIONA EL CURSOR
8260 PRINT "PULSE <ESC> PARA MODIFICAR EL CAMPO,
     <RETURN> PARA SEGUIR"
8270 YPOS=6:XPOS=11
8280 GOSUB 15800:REM -->POSICIONA EL CURSOR
8290 GOSUB 15900:REM -->LECTURA CARACTER
8300 IF ASC(CAR$)<>27 THEN GOTO 8380
8310 YPOS=6:XPOS=13
8320 GOSUB 15800:REM -->POSICIONA EL CURSOR
8330 PRINT "
     "
8340 SW=3
8350 XPOS=13:YPOS=6
8360 GOSUB 1000:REM --->TECLADO
8370 LSET IN$=ALFA$
8380 REM
8390 YPOS=8:XPOS=11
8400 GOSUB 15800:REM -->POSICIONA EL CURSOR
8410 GOSUB 15900:REM -->LECTURA CARACTER
8420 IF ASC(CAR$)<>27 THEN GOTO 8500
8430 YPOS=8:XPOS=13
8440 GOSUB 15800:REM POSICIONA EL CURSOR
```

```
8450 PRINT "
     "
8460 SW=3
8470 GOSUB 15800:REM -->POSICIONA EL CURSOR
8480 GOSUB 1000:REM --->TECLADO
8490 LSET CI$=ALFA$
8500 REM
8510 YPOS=10:XPOS=11
8520 GOSUB 15800:REM -->POSICIONA EL CURSOR
8530 GOSUB 15900:REM LECTURA CARACTER
8540 IF ASC(CAR$)<>27 THEN GOTO 8620
8550 YPOS=10:XPOS=13
8560 GOSUB 15800:REM -->POSICIONA EL CURSOR
8570 PRINT "
     "
8580 SW=2
8590 XPOS=13:YPOS=10
8600 GOSUB 1000:REM --->TECLADO
8610 LSET TE$=MKS$(GRANDE#)
8620 REM
8630 YPOS=12:XPOS=11
8640 GOSUB 15800:REM -->POSICIONA EL CURSOR
8650 GOSUB 15900:REM LECTURA CARACTER
8660 IF ASC(CAR$)<>27 THEN GOTO 8740
8670 YPOS=12:XPOS=13
8680 GOSUB 15800:REM -->POSICIONA EL CURSOR
8690 PRINT "
     "
8700 SW=3
8710 YPOS=12:XPOS=13
8720 GOSUB 1000:REM --->TECLADO
8730 LSET PR$=ALFA$
8740 REM
8750 GOSUB 15500:REM -->ESCRITURA CLIENTE
8760 RETURN
8770 REM
9000 REM ::::::::::::::
9010 REM ::  BORRADO  ::
9020 REM ::::::::::::::
9030 REM
9040 GOSUB 500:REM ---->BORRADO PANTALLA
```



```
9050 YPOS=20:XPOS=1
9060 GOSUB 15800:REM -->POSICIONA EL CURSOR
9070 INPUT "CLAVE A BUSCAR:";NOMINATIVO$
9080 IF LEN(NOMINATIVO$)=31 THEN GOTO 9100
9090 NOMINATIVO$=NOMINATIVO$+" ":GOTO 9080
9100 REM
9110 GOSUB 13000:REM -->CLAVE
9120 IF FIN=1 THEN GOTO 9250
9130 GOSUB 10000:REM -->MASCARA
9140 GOSUB 11000:REM -->VISUALIZACION
9150 YPOS=20:XPOS=1
9160 GOSUB 15800:REM -->POSICIONA EL CURSOR
9170 PRINT "¿QUIERE BORRARLO?(S/N)"
9180 GOSUB 15900:REM -->LECTURA CARACTER
9190 IF CAR$="S" OR CAR$="s" THEN GOTO 9210
9200 RETURN
9210 REM
9220 LSET TI$="C"
9230 GOSUB 15500:REM -->ESCRITURA CLIENTE
9240 RETURN
9250 REM
9260 YPOS=20:XPOS=1
9270 INPUT "NO EXISTE ESTE CLIENTE.PULSE
     <RETURN>";RESPUESTA$
9280  RETURN
10000 REM ::::::::::::::
10010 REM ::  MASCARA  ::
10020 REM ::::::::::::::
10030 REM
10040 YPOS=4:XPOS=1
10050 GOSUB 15800:REM -->POSICIONA EL CURSOR
10060 PRINT "CLIENTE    :"
10070 YPOS=6:XPOS=1
10080 GOSUB 15800:REM -->POSICIONA EL CURSOR
10090 PRINT "DIRECCION  :"
10100 YPOS=8:XPOS=1
10110 GOSUB 15800:REM -->POSICIONA EL CURSOR
10120 PRINT "CIUDAD     :"
10130 YPOS=10:XPOS=1
10140 GOSUB 15800:REM -->POSICIONA EL CURSOR
10150 PRINT "TELEFONO   :"
```

```
10160 YPOS=12:XPOS=1
10170 GOSUB 15800:REM -->POSICIONA EL CURSOR
10180 PRINT "PREFIJO    :"
10190 YPOS=14:XPOS=1
10200 GOSUB 15800:REM -->POSICIONA EL CURSOR
12500 RETURN
10260 REM
11000 REM ::::::::::::::::::::
11010 REM ::  VISUALIZACION  ::
11020 REM ::::::::::::::::::::
11030 REM
11040 YPOS=4:XPOS=13
11050 GOSUB 15800:REM -->POSICIONA EL CURSOR
11060 PRINT CL$
11070 IF TI$<>" " THEN PRINT " BORRADO";
11080 YPOS=6:XPOS=13
11090 GOSUB 15800:REM -->POSICIONA EL CURSOR
11100 PRINT IN$
11110 YPOS=8:XPOS=13
11120 GOSUB 15800:REM -->POSICIONA EL CURSOR
11130 PRINT CI$
11140 YPOS=10:XPOS=13
11150 GOSUB 15800:REM -->POSICIONA EL CURSOR
11160 TELEFONO#=CVS(TE$)
11170 PRINT TELEFONO#
11180 YPOS=12:XPOS=13
11190 GOSUB 15800:REM -->POSICIONA EL CURSOR
11200 PRINT PR$
11290 RETURN
11300 REM
11500 REM ::::::::::::
11510 REM ::  LISTA  ::
11520 REM ::::::::::::
11530 REM
11540 FIN=0
11550 INICIO%>=ACTUAL% OR INICIO%<=0 THEN
      GOTO 11610
11560 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,
      4 AS TE$,4 AS PR$,1 AS TI$
11570 NUMERO%=INICIO%
11580 GOSUB 15000:REM -->LECTURA CLIENTE
```



```
11590 INICIO%=INICIO%+1
11600 RETURN
11610 FIN=1
11620 RETURN
11800 REM ::::::::::::::::
11810 REM ::  VE CLIENTE  ::
11820 REM ::::::::::::::::
11830 REM
11840 FOR NUMERO%=2 TO ACTUAL%-1
11850 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,
      4 AS TE$,4 AS PR$,1 AS TI$
11860 GOSUB 15000:REM -->LECTURA CLIENTE
11870 GOSUB 500:REM ---->BORRADO PANTALLA
11880 GOSUB 10000:REM -->MASCARA
11890 GOSUB 11000:REM -->VISUALIZACION
11900 INPUT "PULSE <RETURN> PARA SEGUIR";RESPUESTA$
11910 NEXT NUMERO%
11920 RETURN
12000 REM ::::::::::::::
12010 REM ::  ACT_NUDO  ::
12020 REM ::::::::::::::
12030 REM
12040 HECHO=0
12050 IF ACTUAL%=2 THEN LSET SI$=MKI$(0):LSET DE$=
      MKI$(0):LSET NO$=NOMINATIVO$:PUT#2,2:GOTO 12520
12060 REM
12070 REM
12080 NUMERO%=P%
12090 GOSUB 16000:REM -->LECTURA INDICE
12100 IF NOMINATIVO$>NO$ THEN GOTO 12300
12110 IF CVI(SI$)<>0 THEN P%=CVI(SI$):GOTO 12500
12120 HECHO=1
12130 LSET SI$=MKI$(ACTUAL%)
12140 GOSUB 16500:REM -->ESCRITURA INDICE
12145 NUMERO%=ACTUAL%
12146 GOSUB 16000:REM -->LECTURA INDICE
12147 LSET NO$=NOMINATIVO$
12150 LSET SI$=MKI$(0)
12160 LSET DE$=MKI$(-P%)
12165 GOSUB 16500:REM -->ESCRITURA INDICE
12170 GOTO 12500
```

```
12300 REM
12310 IF CVI(DE$)>0 THEN P%=CVI(DE$):GOTO 12500
12320 HECHO=1
12330 M%=CVI(DE$)
12340 LSET DE$=MKI$(ACTUAL%)
12350 GOSUB 16500
12355 NUMERO%=ACTUAL%
12356 GOSUB 16000:REM -->LECTURA INDICE
12360 LSET NO$=NOMINATIVO$
12365 LSET DE$=MKI$(M%)
12370 LSET SI$=MKI$(0)
12380 GOSUB 16500:REM -->ESCRITURA INDICE
12500 REM
12510 IF HECHO=0 THEN GOTO 12080
12520 REM
12530 ACTUAL%=ACTUAL%+1
12540 RETURN
12550 REM
13000 REM :::::::::::::
13010 REM ::  CLAVE  ::
13020 REM :::::::::::::
13030 REM
13040 ENC=0
13050 FIN=0
13060 ESTO%=12
13070 NUMERO%=2
13080 IF LEN(NOMINATIVO$)>=31 THEN GOTO 13078
13077 NOMINATIVO$=NOMINATIVO$+" ":GOTO 13075
13078 REM
13080 IF NUMERO%=0 THEN GOTO 13200
13090 GOSUB 16000:REM -->LECTURA INDICE
13100 ESTO%=NUMERO%
13110 IF NOMINATIVO$<NO$ THEN NUMERO%=CVI(SI$):
    GOTO 13080
13120 IF NOMINATIVO$=NO$ THEN ENC=1:PROXIMO%=
     CVI(DE$):GOTO 13300
13130 IF CVI(DE$)>0 THEN NUMERO%=CVI(DE$):GOTO
     13080
13140 NUMERO%=0
13200 FIN=1
13210 PROXIMO%=ESTO%
```



```
13220 RETURN
13300 IF FIN=1 THEN RETURN
13305 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,
     4 AS TE$,4 AS PR$,1 AS TI$
13310 GOSUB 15000:REM -->LECTURA CLIENTE
13320 RETURN
13330 REM
13500 REM :::::::::::::::
13510 REM ::  BUSQUEDA  ::
13520 REM :::::::::::::::
13530 REM
13540 FUERA=0
13550 REM
13560 ESTO%=ABS(PROXIMO%)
13565 NUMERO%=ESTO%
13570 GOSUB 15000:REM -->LECTURA CLIENTE
13575 GOSUB 16000:REM -->LECTURA INDICE
13580 IF CVI(SI$)<>0 AND PROXIMO%>=0 THEN
     PROXIMO%=CVI(SI$):GOTO 13800
13590 FUERA=1
13600 PROXIMO%=CVI(DE$)
13800 IF FUERA=0 THEN GOTO 13550
13810 RETURN
15000 REM ::::::::::::::::::::::::::::::
15010 REM ::  LECTURA FICHERO CLIENTE  ::
15020 REM ::::::::::::::::::::::::::::::
15030 REM
15040 GET #1,NUMERO%
15050 RETURN
15090 REM
15500 REM ::::::::::::::::::::::::::::::::
15510 REM ::  ESCRITURA FICHERO CLIENTE  ::
15520 REM ::::::::::::::::::::::::::::::::
15530 REM
15550 PUT #1,NUMERO%
15560 RETURN
15570 REM
15800 REM ::::::::::::::::::::::::::
15810 REM ::  POSICIONA EL CURSOR  ::
15820 REM ::::::::::::::::::::::::::
15830 REM
```

```
15840 LOCATE YPOS,XPOS,1
15850 RETURN
15860 REM
15900 REM ::::::::::::::::::::::::::::
15910 REM ::  LECTURA  UN CARACTER  ::
15920 REM ::::::::::::::::::::::::::::
15930 REM
15940 CAR$=INPUT$(1)
15950 RETURN
15960 REM
16000 REM ::::::::::::::::::::::
16010 REM ::  LECTURA INDICE  ::
16020 REM ::::::::::::::::::::::
16030 REM
16040 GET #2,NUMERO%
16050 RETURN
16060 REM
16500 REM :::::::::::::::::::::::
16510 REM ::  ESCRITURA INDICE  ::
16520 REM :::::::::::::::::::::::
16530 REM
16540 PUT #2,NUMERO%
16550 RETURN
16560 REM
```

IS_COMPACTA

```
100 REM ******************
110 REM **  IS_COMPACTA  **
120 REM ******************
130 REM
140 OPEN "R",2,"CLIENTES.DAT",88
150 OPEN "R",1,"CLIENTES.VEC",88
170 FIELD 2,2 AS ACTUAL$,2 AS ULTIMO$
175 GET #2,1
180 FIELD 1,2 AS ATC$,2 AS ULT$
190 GET #1,1
200 OPEN "R",3,"INDICE.DAT",35
205 FIELD 3,31 AS NO$,2 AS SI$,2 AS DE$
210 X%=CVI(ULTIMO$)
```



```
220 Y%=CVI(ATC$)
230 NUMERO%=1
235 GOSUB 15500:REM -->ESCRITURA CLIENTE
240 ACTUAL%=2
250 Z%=2
260 REM
270 FIELD 1,31 AS CLI$,33 AS IND$,15 AS CIT$,
    4 AS TEL$,4 AS PRE$,1 AS TIP$
280 GET #1,Z%
290 IF TIF$<>"C" THEN GOTO 320
300 PRINT CLI$,"ANULADO"
310 GOTO 475
320 REM
330 IF Z%>X$+1 THEN GOTO 500
340 NOMINATIVO$=CLI$
345 FIELD 2,31 AS CL$,33 AS IN$,15 AS CI$,
    4 AS TE$,4 AS PR$,1 AS TI$
360 LSET CL$=CLI$
370 LSET IN$=IND$
380 LSET CI$=CIT$
390 LSET TE$=TEL$
400 LSET PR$=PRE$
410 LSET TI$=TIP$
450 PRINT CL$
460 NUMERO%=ACTUAL%
465 GOSUB 15500:REM -->ESCRITURA CLIENTE
470 GOSUB 12000:REM -->ACT_NUDO
475 Z%=Z%+1
480 IF Z%<Y%+1 THEN GOTO 260
490 GOTO 600
500 REM
510 PRINT "HAY",Z%,"REGISTROS ACTIVOS EN UN
    FICHERO QUE LOS PREVEE",X%,"!"
520 PRINT "NO PUEDO SEGUIR."
530 INPUT "PULSE <RETURN>";RESPUESTA$
540 GOTO 650
600 REM
610 FIELD 2,2 AS ACT$,2 AS ULT$
620 NUMERO%=1
625 GOSUB 15000:REM -->LECTURA CLIENTE
630 LSET ACT$=MKI$(ACTUAL%)
```

```
640 GOSUB 15500:REM -->ESCRITURA CLIENTE
650 CLOSE 1
660 CLOSE 2
670 CLOSE 3
680 END
12000 REM ::::::::::::::::
12010 REM ::  ACT_NUDO  ::
12020 REM ::::::::::::::::
12030 REM
12040 HECHO=0
12050 IF ACTUAL%=2 THEN LSET SI$=MKI$(0):LSET DE$=
      MKI$(0):LSET NO$=NOMINATIVO$:PUT #3,2:GOTO 12520
12060 P%=2
12070 REM
12080 NUMERO%=P%
12090 GOSUB 16000:REM -->LECTURA INDICE
12100 IF NOMINATIVO$>NO$ THEN GOTO 12300
12110 IF CVI(SI$)<>0 THEN P%=CVI(SI$):GOTO 12500
12120 HECHO=1
12130 LSET SI$=MKI$(ACTUAL%)
12140 GOSUB 16500:REM -->ESCRITURA INDICE
12145 NUMERO%=ACTUAL%
12146 GOSUB 16000:REM -->LECTURA INDICE
12147 LSET NO$=NOMINATIVO$
12150 LSET SI$=MKI$(0)
12160 LSET DE$=MKI$(-P%)
12165 GOSUB 16500:REM -->ESCRITURA INDICE
12170 GOTO 12500
12300 REM
12310 IF CVI(DE$)>0 THEN P%=CVI(DE%):GOTO 12500
12320 HECHO=1
12330 M%=CVI(DE$)
12340 LSET DE$=MKI$(ACTUAL%)
12350 GOSUB 16500:REM -->ESCRITURA INDICE
12355 NUMERO%=ACTUAL%
12356 GOSUB 16000:REM -->LECTURA INDICE
12357 LSET NO$=NOMINATIVO$
12360 LSET DE$=MKI$(M%)
12370 LSET SI$=MKI$(0)
12380 GOSUB 16500:REM -->ESCRITURA INDICE
12500 REM
```



```
12510 IF HECHO=0 THEN GOTO 12080
12520 REM
12530 ACTUAL%=ACTUAL%+1
12540 RETURN
12550 REM
15000 REM ::::::::::::::::::::::::::::::::
15010 REM ::  LECTURA FICHERO CLIENTE  ::
15020 REM ::::::::::::::::::::::::::::::::
15030 REM
15040 GET #2,NUMERO%
15050 RETURN
15060 REM
15500 REM ::::::::::::::::::::::::::::::::::
15510 REM ::  ESCRITURA FICHERO CLIENTE  ::
15520 REM ::::::::::::::::::::::::::::::::::
15530 REM
15540 PUT #2,NUMERO%
15550 RETURN
15560 REM
16000 REM ::::::::::::::::::::::::::::::::
16010 REM ::  LECTURA FICHERO INDICE  ::
16020 REM ::::::::::::::::::::::::::::::::
16030 REM
16040 GET #3,NUMERO%
16050 RETURN
16060 REM
16500 REM ::::::::::::::::::::::::::::::::
16510 REM :: ESCRITURA FICHERO INDICE  ::
16520 REM ::::::::::::::::::::::::::::::::
16530 REM
16540 PUT #3,NUMERO%
16550 RETURN
```

HA_PREPARA

```
100 REM *******************
110 REM **  HASH_PREPARA  **
120 REM *******************
130 REM
140 FICHERO$="CLIENTE.DAT"
```

```
150 DIVISOR=7
160 OPEN "R",1,FICHERO$,90
170 INPUT "¿DE CUANTOS REGISTROS ESTARA COMPUESTO
    EL FICHERO?";ULTIMO$
180 PRIMERO=2
190 FIELD 1,2 AS PRIMERO$,2 AS ULTIMO$
200 LSET PRIMERO$=MKI$(PRIMERO)
210 LSET ULTIMO$=MKI$(ULTIMO+1)
220 PUT #1,1
230 FIELD 1,31 AS CLIENTE$,33 AS DIRECCION$,15 AS
    CIUDAD,4 AS TELEFONO$,4 AS PREFIJO,1 AS COLISION$
250 LSET CLIENTE$=""
260 LSET DIRECCION$=""
270 LSET CIUDAD$=""
280 LSET TELEFONO$="0"
290 LSET PREFIJO$="0000"
300 LSET TIPO$=" "
310 LSET COLISION$=MKI$(-1)
320 FOR X%=2 TO ULTIMO+1
330 PUT #2,X%
340 NEXT X%
350 CLOSE #1
400 REM FICHERO INDICE
410 FICHERO$="INDICE.DAT"
420 OPEN "R",2,FICHERO$,2
430 FIELD 2,2 AS INDICE$
435 INDICE%=-1
440 LSET INDICE$=MKI$(INDICE%)
450 FOR X=1 TO DIVISOR
460 PUT #2,2
470 NEXT X
480 CLOSE #2
490 END
```

HA_GESTION

```
100 REM ********************
110 REM **  HASH_GESTION  **
120 REM ********************
130 REM
```



```
140 OPEN "R",1,"CLIENTES.DAT",90
150 OPEN "R",2,"INDICE.DAT",2
160 DIVISOR=7
170 NUMERO%=1
180 FIELD 1,2 AS ACTUAL$,2 AS ULTIMO$
190 GOSUB 15000:REM -->LECTURA CLIENTE/1
200 ACTUAL%=CVI(ACTUAL$)
210 ULTIMO%=CVI(ULTIMO$)
220 GOSUB 500:REM ---->BORRADO PANTALLA
230 GOSUB 800:REM ---->MENU
240 ON QUEQUIERE GOSUB 2000,4000,6000,8000,9000
250 IF QUEQUIERE<>9 GOTO 220
270 CLOSE 1
280 CLOSE 2
290 END
500 REM ::::::::::::::::::::::::
510 REM ::  BORRADO PANTALLA  ::
520 REM ::::::::::::::::::::::::
530 REM
540 CLS
550 RETURN
800 REM ::::::::::::
810 REM ::  MENU  ::
820 REM ::::::::::::
830 REM
840 PRINT "1 --> CARGA DE DATOS"
850 PRINT
860 PRINT "2 --> LECTURA CON EL NUMERO DE REGISTRO"
870 PRINT
880 PRINT "3 --> LECTURA CON EL NOMBRE DEL CLIENTE"
890 PRINT
900 PRINT "4 --> MODIFICACION DE REGISTRO"
910 PRINT
920 PRINT "5 --> BORRADO DE UN REGISTRO"
930 PRINT
940 PRINT "9 --> FIN DEL PROGRAMA"
950 PRINT
960 PRINT "?":RESPUESTA$=INPUT$(1):QUEQUIERE=
    ASC(RESPUESTA$)-ASC("0")
980 RETURN
1000 REM :::::::::::::::
```

```
1010 REM ::  TECLADO  ::
1020 REM ::::::::::::::
1030 REM
1040 ALFA$=""
1050 GOSUB 15800:REM -->POSICIONA EL CURSOR
1060 GOSUB 15900:REM -->LECTURA CARACTER
1070 IF ASC(CAR$)=13 THEN GOTO 1210
1080 IF ASC(CAR$)<>8 THEN GOTO 1310
1090 REM ------------------------
1100 REM --  TECLA DE RETORNO  --
1110 REM ------------------------
1120 IF LEN(ALFA$)>1 THEN ALFA$=LEFT$(ALFA$,
     LEN(ALFA$)-1) ELSE ALFA$=""
1130 REM ---------------------------
1140 REM --  IMPRESION DEL CAMPO  --
1150 REM ---------------------------
1160 GOSUB 15800:REM -->POSICIONA EL CURSOR
1170 PRINT ALFA$;" ";
1180 GOSUB 15800:REM -->POSICIONA EL CURSOR
1190 PRINT ALFA$
1200 GOTO 1060
1210 REM ---------------------
1220 REM --  NUEVA ENTRADA  --
1230 REM ---------------------
1240 IF SW=3 THEN RETURN
1250 G#=0
1260 FOR X=1 TO LEN(ALFA$)
1270 G#=G#*10+ASC(MID$(ALFA$,X,1))-48
1280 NEXT X
1290 IF SW=1 THEN PEQUENO%=CSNG(G#) ELSE GRANDE#=G#
1300 RETURN
1310 REM -------------------------
1320 REM --  CARACTER NORMALES  --
1330 REM -------------------------
1340 IF SW=3 THEN  GOTO 1450
1350 REM -------------------------
1360 REM --  CARACTER NUMERICO  --
1370 REM -------------------------
1380 IF CAR$<"0" OR CAR$>"9" THEN GOTO 1400
1390 GOTO 1490
1400 REM ----------------------
```



```
1410 REM --  ERROR DE TECLA  --
1420 REM ----------------------
1430 PRINT CHR$(7);
1440 GOTO 1060
1450 REM --------------------------
1460 REM --  CARACTER ALFABETICO  --
1470 REM --------------------------
1480 IF CAR$<" " OR CAR$>"z" THEN GOTO 1400
1490 ALFA$=ALFA$+CAR$
1500 GOTO 1130
2000 REM ::::::::::::::::::
2010 REM ::  CARGA DATOS  ::
2020 REM ::::::::::::::::::
2030 REM
2040 GOSUB 500:REM ---->BORRADO PANTALLA
2050 PRINT "LA CARGA PARTE CON EL NUMERO:"ACTUAL%-1
2060 IF ACTUAL%<3 THEN GOTO 2140
2070 NUMERO%=ACTUAL%-1
2080 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,
     4 AS TE$,4 AS PR$,1 AS TI$,2 AS CO$
2090 GOSUB 15000:REM -->LECTURA CARACTER
2100 PRINT "EL REGISTRO ANTERIOR ESCRITO ES EL
     SIGUIENTE:"
2110 GOSUB 10000:REM -->MASCARA
2120 GOSUB 11000:REM -->VISUALIZACION
2130 PRINT
2140 INPUT "PULSE <RETURN> PARA INICIAR LA CARGA";
     RESPUESTA$
2150 REM
2160 GOSUB 500:REM ---->BORRADO PANTALLA
2170 YPOS=20:XPOS=1
2180 GOSUB 15800:REM -->POSICIONA EL CURSOR
2190 PRINT "PARA ACABAR LA CARGA RESPONDA 'zzzz'
     A LA PETICION DEL NOMBRE DEL CLIENTE"
2200 GOSUB 10000:REM -->MASCARA
2210 XPOS=13:YPOS=4
2220 SW=3
2230 GOSUB 1000:REM --->TECLADO
2240 NOMINATIVO$=ALFA$
2250 IF NOMINATIVO$="zzzz" THEN RETURN
2260 IF ACTUAL%>ULTIMO% THEN INPUT "NO PUEDO
```

```
     SEGUIR.SE HA SUPERADO LA DIMENSION MAXIMA
     DEL FICHERO.PULSE <RETURN>";RESPUESTA$:RETURN
2270 GOSUB 12000:REM -->CALCULO
2280 GOSUB 13000:REM -->BUSQUEDA
2290 IF FIN=1 THEN GOTO 2320
2300 INPUT "ESTE CLIENTE YA EXISTE.NO PUEDO
     INTRODUCIRLO.PULSE <RETURN>";RESPUESTA$
2310 RETURN
2320 CLIENTE$=NOMINATIVO$
2330 SW=3
2340 XPOS=13
2350 YPOS=6
2360 GOSUB 1000:REM -->TECLADO
2370 DIRECCION$=ALFA$
2380 SW=3
2390 XPOS=13
2400 YPOS=9
2410 GOSUB 1000:REM --->TECLADO
2420 CIUDAD$=ALFA$
2430 SW=2
2440 XPOS=13
2450 YPOS=10
2460 GOSUB 1000:REM --->TECLADO
2470 TELEFONO#=GRANDE#
2480 SW=3
2490 XPOS=13
2500 YPOS=12
2510 GOSUB 1000:REM --->TECLADO
2520 PREFIJO$=ALFA$
2530 TIPO$=" "
2540 GOSUB 15200:REM -->LECTURA HASH
2550 COLISION$=CVI(HASH$)
2560 LSET HASH$=MKI$(ACTUAL%)
2570 GOSUB 15700:REM -->ESCRITURA HASH
2580 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,
     4 AS TE$,4 AS PR$,1 AS TI$,2 AS CO$
2590 LSET CL$=CLIENTE$
2600 LSET IN$=DIRECCION$
2610 LSET CI$=CIUDAD$
2620 LSET TE$=MKS$(TELEFONO#)
2630 LSET PR$=PREFIJO$
```



```
2640 LSET TI$=TIPO$
2650 LSET CO$=MKI$(COLISION#)
2660 NUMERO%=ACTUAL%
2670 GOSUB 15500:REM -->ESCRITURA CLIENTE
2680 ACTUAL%=ACTUAL%+1
2690 FIELD 1,2 AS ACTUAL$,2 AS ULTIMO$
2700 LSET ACTUAL$=MKI$(ACTUAL%)
2710 LSET ULTIMO$=MKI$(ULTIMO%)
2720 NUMERO%=1
2730 GOSUB 15500:REM -->ESCRITURA CLIENTE/1
2740 GOTO 2150
4000 REM :::::::::::::::::::::::::::::::::::::::::
4010 REM ::  LECTURA CON EL NUMERO DE REGISTRO  ::
4020 REM :::::::::::::::::::::::::::::::::::::::::
4030 REM
4040 YPOS=20:XPOS=1
4050 GOSUB 15800:REM -->POSICIONA EL CURSOR
4060 PRINT "NUMERO DEL PRIMER REGISTRO:"
4070 SW=1
4080 XPOS=32
4090 YPOS=20
4100 GOSUB 1000:REM --->TECLADO
4110 INICIO%=PEQUENO%+1
4120 YPOS=20:XPOS=1
4130 GOSUB 15800:REM -->POSICIONA EL CURSOR
4140 PRINT "NUMERO DEL ULTIMO REGISTRO:    "
4150 SW=1
4160 XPOS=32
4170 YPOS=20
4180 GOSUB 1000:REM --->TECLADO
4190 ALT%=PEQUENO%+1
4200 IF ALT%>ACTUAL%-1 THEN ALT%=ACTUAL%-1
4210 GOSUB 11500:REM -->LISTA
4220 IF FIN=1 THEN RETURN
4230 REM
4240 GOSUB 500:REM ---->BORRADO PANTALLA
4250 GOSUB 10000:REM -->MASCARA
4260 GOSUB 11000:REM -->VISUALIZACION
4270 YPOS=20:XPOS=1
4280 GOSUB 15800:REM -->POSICIONA EL CURSOR
4290 GOSUB 11500:REM -->LISTA
```

```
4300 PRITN "<ESPACIO> PARA SEGUIR,<ESC> PARA
     TERMINAR"
4310 RESPUESTA$=INPUT$(1)
4320 IF ASC(RESPUESTA$)=27 OR FIN=1 OR INICIO%>
     ALT%+1 THEN RETURN
4330 GOTO 4230
6000 REM ::::::::::::::
6010 REM ::  NOMBRE  ::
6020 REM ::::::::::::::
6030 REM
6040 YPOS=20
6050 XPOS=1
6060 GOSUB 15800:REM -->POSICIONA EL CURSOR
6070 PRINT "NOMBRE DE CLIENTE     :"
6080 SW=3
6090 XPOS=32
6100 YPOS=20
6110 GOSUB 1000:REM --->TECLADO
6120 NOMINATIVO$=ALFA$
6125 GOSUB 12000:REM -->CALCULO HASH
6130 IF LEN(NOMINATIVO$)=31 THEN GOTO 6150
6140 NOMINATIVO$=NOMINATIVO$+" ":GOTO 6130
6160 GOSUB 13000:REM -->BISQUEDA
6170 IF FIN=1 THEN GOTO 6250
6180 GOSUB 500:REM ---->BORRA PANTALLA
6190 GOSUB 10000:REM -->MASCARA
6200 GOSUB 11000:REM -->VISUALIZACION
6210 XPOS=1:YPOS=20
6220 GOSUB 15800:REM -->POSICIONA EL CURSOR
6230 INPUT "PULSA <RETURN> PARA SEGUIR";RESPUESTA$
6240 RETURN
6250 REM
6260 YPOS=20:XPOS=1
6270 GOSUB 15800:REM -->POSICIONA EL CURSOR
6280 INPUT "NO EXISTE ESTE CLIENTE.PULSE <RETURN>";
     RESPUESTA$
6290 RETURN
8000 REM ::::::::::::::::::::
8010 REM ::  MODIFICACION  ::
8020 REM ::::::::::::::::::::
8030 REM
```



```
8040 GOSUB 500:REM ---->BORRADO PANTALLA
8050 YPOS=20:XPOS=1
8060 GOSUB 15800:REM -->POSICIONA EL CURSOR
8070 PRINT "CLAVE DE BUSQUEDA:"
8080 SW=3
8090 XPOS=23:YPOS=20
8100 GOSUB 1000:REM --->TECLADO
8110 NOMINATIVO$=ALFA$
8120 GOSUB 12000:REM -->CALCULO
8130 IF LEN(NOMINATIVO%)=31 THEN GOTO 8150
8140 NOMINATIVO$=NOMINATIVO$+" ":GOTO 8130
8150 GOSUB 13000:REM -->BUSQUEDA
8160 IF FIN=0 THEN GOTO 8210
8170 YPOS=20:XPOS=1
8180 GOSUB 15800:REM -->POSICIONA EL CURSOR
8190 INPUT "NO EXISTE ESTE CLIENTE.PULSE <RETURN>";
     RESPUESTA$
8200 RETURN
8210 REM
8220 GOSUB 10000:REM -->MASCARA
8230 GOSUB 11000:REM VISUALIZACION
8240 YPOS=20:XPOS=1
8250 GOSUB 15800:REM -->POSICIONA EL CURSOR
8260 PRINT "PULSE <ESC> PARA MODIFICAR EL CAMPO,
     <RETURN> PARA SEGUIR"
8270 YPOS=6:XPOS=11
8280 GOSUB 15800:REM -->POSICIONA EL CURSOR
8290 GOSUB 15900:REM -->LECTURA CARACTER
9300 IF ASC(CAR$)<>27 THEN GOTO 8380
8310 YPOS=6:XPOS=13
8320 GOSUB 15800:REM -->POSICIONA EL CURSOR
8330 PRINT "
     "
8340 SW=3
8350 XPOS=13:YPOS=6
8360 GOSUB 1000:REM --->TECLADO
8370 LSET IN$=ALFA$
8380 REM
8390 YPOS=8:XPOS=11
8400 GOSUB 15800:REM -->POSICIONA EL CURSOR
8410 GOSUB 15900:REM -->LECTURA CARACTER
```

```
8420 IF ASC(CAR$)<>27 THEN GOTO 8500
8430 YPOS=8:XPOS=13
8440 GOSUB 15800:REM -->POSICIONA EL CURSOR
8450 PRINT "
     "
8460 SW=3
8470 GOSUB 15800:REM -->POSICIONA EL CURSOR
8480 GOSUB 1000:REM --->TECLADO
8490 LSET CI$=ALFA$
8500 REM
8510 YPOS=10:XPOS=11
8520 GOSUB 15800:REM -->POSICIONA EL CURSOR
8530 GOSUB 15900:REM -->LECTURA CARACTER
8540 IF ASC(CAR$)<>27 THEN GOTO 8620
8550 YPOS=10:XPOS=13
8560 GOSUB 15800:REM -->POSICIONA EL CURSOR
8570 PRINT "
     "
8580 SW=2
8590 XPOS=13:YPOS=10
8600 GOSUB 1000:REM --->TECLADO
8610 LSET TE$=MKS$(GRANDE#)
8620 REM
8630 YPOS=12:XPOS=11
8640 GOSUB 15800:REM -->POSICIONA EL CURSOR
8650 GOSUB 15900:REM -->LECTURA CARACTER
8660 IF ASC(CAR$)<>27 THEN GOTO 8740
8670 YPOS=12:XPOS=13
8680 GOSUB 15800:REM -->POSICIONA EL CURSOR
8690 PRINT "
     "
8700 SW=3
8710 YPOS=12:XPOS=13
8720 GOSUB 1000:REM --->TECLADO
8730 LSET PR$=ALFA$
8740 REM
8750 GOSUB 15800:REM -->POSICIONA EL CURSOR
8760 RETURN
9000 REM ::::::::::::::::
9010 REM ::  ANULACION  ::
9020 REM ::::::::::::::::
```



```
9030 REM
9040 GOSUB 500:REM ---->BORRADO PANTALLA
9050 YPOS=20:XPOS=1
9060 GOSUB 15800:REM -->POSICIONA EL CURSOR
9070 INPUT "CLAVE A BUSCAR:";NOMINATIVO$
9080 IF LEN(NOMINATIVO$)=31 THEN GOTO 9100
9090 NOMINATIVO$=NOMINATIVO$+" ":GOTO 9080
9100 GOSUB 12000:REM -->CALCULO
9110 GOSUB 13000:REM -->BUSQUEDA
9120 IF FIN=1 THEN GOTO 9250
9130 GOSUB 10000:REM -->MASCARA
9140 GOSUB 11000:REM -->VISUALIZACION
9150 YPOS=20:XPOS=1
9160 GOSUB 15800:REM -->POSICIONA EL CURSOR
9170 PRINT "¿QUIERE BORRARLO?(S/N)":
9180 GOSUB 15900:REM -->LECTURA CARACTER
9190 IF CAR$="S" OR CAR$="s" THEN GOTO 9210
9200 RETURN
9210 REM
9220 LSET TI$="C"
9230 GOSUB 15500:REM -->ESCRITURA CLIENTE
9240 RETURN
9250 REM
9260 YPOS=20:XPOS=1
9270 INPUT "NO EXISTE ESTE CLIENTE.PULSE
     <RETURN>";RESPUESTA$
9280 RETURN
10000 REM ::::::::::::::
10010 REM ::  MASCARA  ::
10020 REM ::::::::::::::
10030 REM
10040 YPOS=6:XPOS=1
10050 GOSUB 15800:REM -->POSICIONA EL CURSOR
10060 PRINT "CLIENTE   :"
10070 YPOS=6:XPOS=1
10080 GOSUB 15800:REM -->POSICIONA EL CURSOR
10090 PRINT "DIRECCION :"
10100 YPOS=8:XPOS=1
10110 GOSUB 15800:REM -->POSICIONA EL CURSOR
10120 PRINT "CIUDAD    :"
10130 YPOS=10:XPOS=1
```

```
10140 GOSUB 15800:REM -->POSICIONA EL CURSOR
10150 PRINT "TELEFONO  :"
10160 YPOS=12:XPOS=1
10170 GOSUB 15800:REM -->POSICIONA EL CURSOR
10180 PRINT "PREFIJO   :"
10190 YPOS=14:XPOS=1
10200 GOSUB 15800:REM -->POSICIONA EL CURSOR
10210 PRINT "COLISION  :"
10220 RETURN
11000 REM ::::::::::::::::::::
11010 REM ::  VISUALIZACION  ::
11020 REM ::::::::::::::::::::
11030 REM
11040 YPOS=4:XPOS=13
11050 GOSUB 15800:REM -->POSICIONA EL CURSOR
11060 PIRNT CL$
11070 IF TI$<>" " THEN PRINT "BORRADO"
11080 YPOS=6:XPOS=13
11090 GOSUB 15800:REM -->POSICIONA EL CURSOR
11100 PRINT IN$
11110 YPOS=8:XPOS=13
11120 GOSUB 15800:REM -->POSICIONA EL CURSOR
11130 PRINT CI$
11140 YPOS=10:XPOS=13
11150 GOSUB 15800:REM -->POSICIONA EL CURSOR
11160 TELEFONO#=CVI(TE$)
11170 PRINT TELEFONO#
11180 YPOS=12:XPOS=13
11190 GOSUB 15800:REM -->POSICIONA EL CURSOR
11200 PRINT PR$
11210 YPOS=14:XPOS=13
11220 GOSUB 15800:REM -->POSICIONA EL CURSOR
11230 COLISION%=CVI(CO$)
11240 PRINT COLISION%
11250 RETURN
11500 REM :::::::::::::
11510 REM ::  LISTA  ::
11520 REM :::::::::::::
11530 REM
11540 FIN=0
11550 IF INICIO%>=ACTUAL% OR INICIO%<=0 THEN GOTO 11610
```



```
11560 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,
      4 AS TE$,4 AS PR$,1 AS TI$,2 AS CO$
11570 NUMERO%=INICIO%
11580 GOSUB 15000:REM -->LECTURA CLIENTE
11590 INICIO%=INICIO%+1
11600 RETURN
11610 FIN=1
11620 RETURN
12000 REM ::::::::::::::::::::::::::
12010 REM ::  CALCULO NUMERO HASH  ::
12020 REM ::::::::::::::::::::::::::
12030 REM
12040 X=0:Y=0
12050 FOR N=1 TO LEN(NOMINATIVO$)
12060 C=ASC(MID$(NOMINATIVO$,N,1))
12070 IF N-(INT(N/2)*2)=0 THEN Y=Y+C ELSE X=X+C
12080 NEXT N
12090 R1=X-(INT(X/256)*256)
12100 R2=Y-(INT(Y/256)*256)
12110 Z=R2*256+R1
12120 HASH=INT(Z-(INT(Z/DIVISOR)*DIVISOR))
12130 IF HASH=0 THEN HASH=DIVISOR
12140 RETURN
13000 REM ::::::::::::::::
13010 REM ::  BUSQUEDA  ::
13020 REM ::::::::::::::::
13030 REM
13040 FIN=0
13050 GOSUB 15200:REM -->LECTURA HASH
13060 CUAL%=CVI(HASH$)
13070 REM
13080 IF CUAL%=-1 THEN FIN=1:RETURN
13090 NUMERO%=CUAL%
13100 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,
      4 AS TE$,4 AS PR$,1 AS TI$,2 AS CO$
13110 GOSUB 15000:REM -->LECTURA CLIENTE
13120 IF NOMINATIVO$=CL$ THEN RETURN
13130 CUAL%=CVI(CO$)
13140 GOTO 13070
15000 REM ::::::::::::::::::::::::::::::
15010 REM ::  LECTURA FICHERO CLIENTE  ::
```

```
15020 REM ::::::::::::::::::::::::::::::
15030 REM
15040 GET #1,NUMERO%
15050 RETURN
15090 REM
15200 REM ::::::::::::::::::::::::::::
15210 REM ::  LECTURA FICHERO HASH  ::
15220 REM ::::::::::::::::::::::::::::
15230 REM
15240 FIELD 2,2 AS HASH$
15250 GET #2,HASH
15260 RETURN
15270 REM
15500 REM ::::::::::::::::::::::::::::::::
15510 REM ::  ESCRITURA FICHERO CLIENTE  ::
15520 REM ::::::::::::::::::::::::::::::::
15530 REM
15550 PUT #1,NUMERO%
15560 RETURN
15590 REM
15700 REM ::::::::::::::::::::::::::::::
15710 REM ::  ESCRITURA FICHERO HASH  ::
15720 REM ::::::::::::::::::::::::::::::
15730 REM
15740 PUT #2,HASH
15750 RETURN
15760 REM
15800 REM ::::::::::::::::::::::::::
15810 REM ::  POSICIONA EL CURSOR  ::
15820 REM ::::::::::::::::::::::::::
15830 REM
15840 LOCATE YPOS,XPOS
15850 RETURN
15860 REM
15900 REM ::::::::::::::::::::::
15910 REM ::  LECTURA CARACTER  ::
15920 REM ::::::::::::::::::::::
15930 REM
15940 CAR$=INPUT$(1)
15950 RETURN
15960 REM
```


HA_COMPACTA

```
100 REM ****************
110 REM *  HA_COMPACTA  *
120 REM ****************
130 REM
140 DIVISOR=7
150 OPEN "R",1,"CLIENTE.VEC",90
160 OPEN "R",2,"INDICE.DAT",2
170 OPEN "R",3,"CLIENTES.DAT",90
180 FIELD 1,2 AS J$,2 AS X$
190 GET #1,1
200 J%=CVI(J$)
210 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,4 AS TE$,4
    AS PR$,1 AS TI$,2 AS CO$
220 FIELD 3,2 AS ACTUAL$,2 AS ULTIMO$
230 GET #3,1
240 ULTIMO%=CVI(ULTIMO$)
250 FIELD 3,31 AS CLI$,33 AS IND$,15 AS CIT$,4 AS
    TEL$,4 AS PRE$,1 AS TIP$,2 AS COL$
260 ACTUAL%=2:K=2:J=2
270 REM
280 GET #1,J
290 IF TI$<>" " THEN PRINT CL$;" BORRADO":J=J+1:
    GOTO 490
300 K=K+1:PRINT CL$
310 NOMINATIVO$=CL$
320 GOSUB 12000:REM -->CALCULO
330 FIELD #2,2 AS HASH$
340 GET #2,HASH
350 HH$=HASH$
360 LSET CO$=HASH$
370 LSET HASH$=MKI$(ACTUAL%)
380 PUT #2,HASH
390 LSET COL$=HH$
400 LSET CLI$=CL$
410 LSET IND$=IN$
420 LSET CIT$=CI$
430 LSET TEL$=TE$
440 LSET PRE$=PR$
450 LSET TIP$=" "
```

```
460 PUT #3,ACTUAL%
470 ACTUAL%=ACTUAL%+1
480 J=J+1
490 REM
500 GET #1,J
510 IF ACTUAL%<ULTIMO% THEN GOTO 560
520 PRINT "HAY ";ACTUAL%-1;" REGISTROS ACTIVOS
    EN UN FICHERO QUE LOS PREVEE ";ULTIMO%-1;"!"
530 PRINT "NO PUEDO SEGUIR"
540 INPUT "PULSE <RETURN>";RESPUESTA$
550 GOTO 620
560 REM
570 IF J<I% THEN GOTO 290
580 FIELD 3,2 AS ACTUAL$,2 AS ULTIMO$
590 LSET ACTUAL$=MKI$(ACTUAL%-1)
600 LSET ULTIMO$=MKI$(ULTIMO%)
610 PUT #3,1
620 REM
630 CLOSE #1
640 CLOSE #2
650 CLOSE #3
660 END
12000 REM ::::::::::::::::::::::::::
12010 REM ::  CALCULO NUMERO HASH  ::
12020 REM ::::::::::::::::::::::::::
12030 REM
12040 X=0:Y=0
12050 FOR N=1 TO LEN(NOMINATIVO$)
12060 C=ASC(MID$(NOMINATIVO$,N,1))
12070 IF N-(INT(N/2)*2)=0 THEN Y=Y+C ELSE X=X+C
12080 NEXT N
12090 R1=X-(INT(X/256)*256)
12100 R2=Y-(INT(Y/256)*256)
12110 Z=R2*256+R1
12120 HASH=INT(Z-(INT(Z/DIVISOR)*DIVISOR))
12130 IF HASH=0 THEN HASH=DIVISOR
12140 RETURN
11610 GOSUB 15000:REM -->LECTURA CLIENTE NUEVO
12530 ACTUAL%=ACTUAL%+1
12540 RETURN
```


AL_PREPARA

```
100 REM ****************
110 REM *  AL_PREPARA  *
120 REM ****************
130 REM
140 FICHERO$="CLIENTES.DAT"
160 OPEN "R",1,FICHERO$,92
170 INPUT "¿DE CUANTOS REGISTROS ESTARA COMPUESTO
    EL FICHERO?";ULTIMO$
180 PRIMERO=2
190 FIELD 1,2 AS PRIMERO$,2 AS ULTIMO$
200 LSET PRIMERO$=MKI$(PRIMERO)
210 LSET ULTIMO$=MKI$(ULTIMO)
220 PUT #1,1
230 FIELD 1,31 AS CLIENTE$,33 AS DIRECCION$,15 AS
    CIUDAD$,4 AS TELEFONO$,4 AS PREFIJO$,1 AS TIPO$,
    2 AS IZQUIERDO$,2 AS DERECHO$
250 LSET CLIENTE$=""
260 LSET DIRECCION$=""
270 LSET CIUDAD$=""
280 LSET TELEFONO$="0"
290 LSET PREFIJO$="0000"
300 LSET TIPO$=" "
310 IZQUIERDO$=MKI$(0)
315 DERECHO$=MKI$(0)
320 FOR X%=2 TO ULTIMO+1
330 PUT #1,X%
340 NEXT X%
350 CLOSE #1
400 END
```

AL_GESTION

```
100 REM ****************
110 REM *  AL_GESTION  *
120 REM ****************
130 REM
140 OPEN "R",1,"CLIENTES.DAT",92
170 NUMERO%=1
```

```
180 FIELD 1,2 AS ACTUAL$,2 AS ULTIMO$
190 GOSUB 15000:REM -->LECTURA CLIENTE/1
200 ACTUAL%=CVI(ACTUAL$)
210 ULTIMO%=CVI(ULTIMO$)
220 GOSUB 500:REM ---->BORRADO PANTALLA
230 GOSUB 800:REM ---->MENU
240 ON QUEQUIERE GOSUB 2000,4000,6000,8000,9000
250 IF QUEQIERE<>9 THEN GOTO 220
270 CLOSE 1
290 END
500 REM ::::::::::::::::::::::::
510 REM ::  BORRADO PANTALLA  ::
520 REM ::::::::::::::::::::::::
530 REM
540 CLS
550 RETURN
800 REM ::::::::::::
810 REM ::  MENU  ::
820 REM ::::::::::::
830 REM
840 PRINT "1 --> CARGA DE DATOS"
850 PRINT
860 PRINT "2 --> LECTURA CON EL NUMERO DE REGISTRO"
870 PRINT
880 PRINT "3 --> LECTURA CON EL NOMBRE DE CLIENTE"
890 PRINT
900 PRINT "4 --> MODIFICACION DE UN REGISTRO"
910 PRINT
920 PRINT "5 --> BORRADO DE UN REGISTRO"
930 PRINT
940 PRINT "9 --> FIN DEL PROGRAMA"
950 PRINT
960 PRINT "?":RESPUESTA$=INPUT$(1):QUEQUIERE=ASC(
    RESPUESTA$)-ASC("0")
980 RETURN
1000 REM ::::::::::::::
1010 REM :: TECLADO  ::
1020 REM ::::::::::::::
1030 REM
1040 ALFA$=""
1050 GOSUB 15800:REM -->POSICIONA EL CURSOR
```



```
1060 GOSUB 15900:REM -->LECTURA CARACTER
1070 IF ASC(CAR$)=13 THEN GOTO 1210
1080 IF ASC(CAR$)<>8 THEN GOTO 1310
1090 REM ------------------------
1100 REM --  TECLA DE RETORNO  --
1110 REM ------------------------
1120 IF LEN(ALFA$)>1 THEN ALFA$=LEFT$(ALFA$,LEN(
     ALFA$)-1) ELSE ALFA$=""
1130 REM --------------------------
1140 REM --  IMPRESION DEL CAMPO  --
1150 REM --------------------------
1160 GOSUB 15800:REM -->POSICIONA EL CURSOR
1170 PRINT ALFA$;" ";
1180 GOSUB 15800:REM -->POSICIONA EL CURSOR
1190 PRINT ALFA$
1200 GOTO 1060
1210 REM ---------------------
1220 REM --  NUEVA ENTRADA  --
1230 REM ---------------------
1240 IF SW=3 THEN RETURN
1250 G#=0
1260 FOR X=1 TO LEN(ALFA$)
1270 G#=G#*10+ASC(MID$(ALFA$,X,1))-48
1280 NEXT X
1290 IF SW=1 THEN PEQUENO%=CSNG(G#) ELSE GRANDE#=G#
1300 RETURN
1310 REM ----------------------
1320 REM --  CARACTER NORMAL  --
1330 REM ----------------------
1340 IF SW=3 THEN GOTO 1450
1350 REM ------------------------
1360 REM --  CARACTER NUMERICO  --
1370 REM ------------------------
1380 IF CAR$<"0" OR CAR$>"9" THEN GOTO 1400
1390 GOTO 1490
1400 REM ----------------------
1410 REM --  ERROR DE TECLA  --
1420 REM ----------------------
1430 PRINT CHR$(7);
1440 GOTO 1060
1450 REM --------------------------
```

```
1460 REM -- CARACTER ALFABETICO --
1470 REM ---------------------------
1480 IF CAR$<" " OR CAR$>"z" THEN GOTO 1400
1490 ALFA$=ALFA$+CAR$
1500 GOTO 1130
2000 REM ::::::::::::::::::::
2010 REM :: BUSQUEDA DATO ::
2020 REM ::::::::::::::::::::
2030 REM
2040 GOSUB 500:REM ---->BORRADO PANTALLA
2050 PRINT "LA CARGA PARTE CON EL NUMERO:";
     ACTUAL%-1
2060 IF ACTUAL%-1<3 THEN GOTO 2140
2070 NUMERO%=ACTUAL%-1
2080 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,4 AS TE$,
     4 AS PR$,1 AS TI$,2 AS SI$,2 AS DE$
2090 GOSUB 15000:REM -->LECTURA CLIENTE
2100 PRINT "EL REGISTRO ANTERIOR ESCRITO ES EL
      SIGUIENTE:"
2110 GOSUB 10000:REM -->MASCARA
2120 GOSUB 11000:REM -->VISUALIZACION
2130 PRINT
2140 INPUT "PULSE <RETURN> PARA INICIAR LA CARGA";
     RESPUESTA$
2150 REM
2155 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,4 AS TE$,
     4 AS PR$,1 AS TI$,2 AS SI$,2 AS DE$
2160 GOSUB 500:REM ---->BORRADO PANTALLA
2170 YPOS=20:XPOS=1
2180 GOSUB 15800:REM -->POSICIONA EL CURSOR
2190 PRINT "PARA FINALIZAR LA CARGA,A LA PETICION
     DEL NOMBRE DEL CLIENTE RESPONDA 'zzzz'"
2200 GOSUB 10000:REM -->MASCARA
2210 XPOS=13:YPOS=4
2220 SW=3
2230 GOSUB 1000:REM --->TECLADO
2240 NOMINATIVO$="zzzz" THEN RETURN
2260 IF ACTUAL%>ULTIMO% THEN INPUT "NO PUEDO SEGUIR
     ,PUES SE HA SUPERADO LA DIMENSION MAXIMA DEL
     FICHERO.PULSE <RETURN>";RESPUESTA$
2280 GOSUB 13000:REM -->CLAVE
```



```
2290 IF ENC=0 THEN GOTO 2310
2296 GOSUB 15800:REM -->POSICIONA EL CURSOR
2300 INPUT "ESTE CLIENTE YA EXISTE.NO PUEDO
     INTRODUCIRLO.PULSE <RETURN>    ";RESPUESTA$
2305 RETURN
2310 GOSUB 12000:REM -->ACT_NUDO
2320 CLIENTE$=NOMINATIVO$
2330 SW=3
2340 XPOS=13
2350 YPOS=6
2360 GOSUB 1000:REM --->TECLADO
2370 DIRECCION$=ALFA$
2380 SW=3
2390 XPOS=13
2400 YPOS=8
2410 GOSUB 1000:REM --->TECLADO
2420 CIUDAD$=ALFA$
2430 SW=2
2440 XPOS=13
2450 YPOS=10
2460 GOSUB 1000:REM --->TECLADO
2470 TELEFONO#=GRANDE#
2480 SW=3
2490 XPOS=13
2500 YPOS=12
2510 GOSUB 1000:REM -->TECLADO
2520 PREFIJO$=ALFA$
2530 TIPO$=" "
2590 LSET CL$=CLIENTE$
2600 LSET IN$=DIRECCION$
2610 LSET CI$=CIUDAD$
2620 LSET TE$=MKS$(TELEFONO#)
2630 LSET PR$=PREFIJO$
2640 LSET TI$=TIPO$
2660 NUMERO%=ACTUAL%
2670 GOSUB 15500:REM -->ESCRITURA CLIENTE
2680 ACTUAL%=ACTUAL%+1
2690 FIELD 1,2 AS ACTUAL$,2 AS ACTUAL$
2700 LSET ACTUAL$=MKI$(ACTUAL%)
2710 LSET ULTIMO$=MKI$(ULTIMO%)
2720 NUMERO%=1
```

```
2730 GOSUB 15500:REM -->ESCRITURA CLIENTE/1
2740 GOTO 2150
4000 REM ::::::::::::::::::::::::::::::::::::::::
4010 REM ::  LECTURA CON EL NUMERO DE REGISTRO  ::
4020 REM ::::::::::::::::::::::::::::::::::::::::
4030 REM
4040 YPOS=20:XPOS=1
4050 GOSUB 15800:REM -->POSICIONA EL CURSOR
4060 PRINT "NUMERO DEL PRIMER REGISTRO:"
4070 SW=1
4080 XPOS=32
4090 YPOS=20
4100 GOSUB 1000:REM --->TECLADO
4110 INICIO%=PEQUENO%+1
4120 YPOS=20:XPOS=1
4130 GOSUB 15800:REM -->POSICIONA EL CURSOR
4140 PRINT "NUMERO DEL ULTIMO REGISTRO"
4150 SW=1
4160 XPOS=32
4170 YPOS=20
4180 GOSUB 1000:REM --->TECLADO
4190 ALT%=PEQUENO%+1
4200 IF ALT%>ACTUAL%-1 THEN ALT%=ACTUAL%-1
4210 GOSUB 11500:REM -->LISTA
4220 IF FIN=1 THEN RETURN
4230 REM
4240 GOSUB 500:REM ---->BORRADO PANTALLA
4250 GOSUB 10000:REM -->MASCARA
4260 GOSUB 11000:REM -->VISUALIZACION
4270 YPOS=20:XPOS=1
4280 GOSUB 15800:REM -->POSICIONA EL CURSOR
4290 GOSUB 11500:REM -->LISTA
4300 PRINT "<ESPACIO> PARA CONTINUAR,<ESC> PARA
     TERMINAR"
4310 RESPUESTA$=INPUT$(1)
4320 IF ASC(RESPUESTA$)=27 OR FIN=1 OR INICIO%>
     ALT%+1 THEN RETURN
4330 GOTO 4230
6000 REM ::::::::::::::
6010 REM ::  NOMBRE  ::
6020 REM ::::::::::::::
```



```
6030 REM
6060 YPOS=20
6070 XPOS=1
6080 GOSUB 15800:REM -->POSICIONA EL CURSOR
6090 PRINT "NOMBRE DEL PRIMER CLIENTE :"
6100 SW=3
6110 XPOS=32
6120 YPOS=20
6130 GOSUB 1000:REM --->TECLADO
6140 PARTIDA$=ALFA$
6150 YPOS=20
6160 XPOS=1
6170 GOSUB 15800:REM -->POSICIONA EL CURSOR
6180 PRINT "NOMBRE DEL ULTIMO CLIENTE :
     "
6190 SW=3
6200 XPOS=32
6210 YPOS=20
6220 GOSUB 1000:REM --->TECLADO
6230 TERMINO$=ALFA$
6231 IF LEN(TERMINO$)>=31 THEN GOTO 6235
6232 TERMINO$=TERMINO$+" ":GOTO 6231
6235 NOMINATIVO$=PARTIDO$
6240 GOSUB 13000:REM -->CLAVE
6250 IF FIN=1 THEN GOTO 6500
6260 GOSUB 500:REM ---->BORRADO PANTALLA
6270 GOSUB 10000:REM -->MASCARA
6280 GOSUB 11000:REM -->VISUALIZACION
6282 ESTO%=PROXIMO%
6282 IF ESTO%=0 THEN GOTO 6290
6287 GOSUB 13500:REM -->BUSQUEDA
6290 YPOS=20:XPOS=1
6300 GOSUB 15800:REM -->POSICIONA EL CURSOR
6310 PRINT "<ESPACIO> PARA CONTINUAR,<ESC>
     PARA TERMINAR:";
6320 GOSUB 15900:REM -->LECTURA CARACTER
6330 IF ASC(CAR$)=27 THEN GOTO 6380
6340 IF CL$>TERMINO$ THEN GOTO 6380
6350 IF ESTO%=0 THEN GOTO 6380
6370 GOTO 6260
6380 REM
```

```
6400 RETURN
6410 REM
6500 REM
6510 GOSUB 13500:REM -->BUSQUEDA
6520 FIN=0
6530 GOTO 6260
6540 REM
8000 REM ::::::::::::::::::::
8010 REM ::  MODIFICACION  ::
8020 REM ::::::::::::::::::::
8030 REM
8040 GOSUB 500:REM ---->BORRADO PANTALLA
8050 YPOS=20:XPOS=1
8060 GOSUB 15800:REM-->POSICIONA EL CURSOR
8070 PRINT "CLAVE A BUSCAR:"
8080 SW=3
8090 XPOS=23:YPOS=20
8100 GOSUB 100:REM --->TECLADO
8110 NOMINATIVO$=ALFA$
8130 IF LEN(NOMINATIVO$)=31 THEN GOTO 8150
8140 NOMINATIVO$=NOMINATIVO$+" ":GOTO 8130
8150 GOSUB 1300:REM -->CLAVE
8160 IF FIN=0 THEN GOTO 8210
8170 YPOS=20:XPOS=1
8180 GOSUB 15800:REM -->POSICIONA EL CURSOR
8190 INPUT "NOEXISTE ESTE CLIENTE.PULSE <RETURN>";
     RESPUESTA$
8200 RETURN
8210 REM
8220 GOSUB 10000:REM -->MASCARA
8230 GOSUB 11000:REM --VISUALIZACION
8240 YPOS=20:XPOS=1
8250 GOSUB 15800:REM -->POSICIONA EL CURSOR
8260 PRINT "PULSE <ESC> PARA MODIFICAR EL CAMPO,
     <RETURN> PARA SEGUIR"
8270 YPOS=6:XPOS=11
8280 GOSUB 15800:REM -->POSICIONA EL CURSOR
8290 GOSUB 15900:REM -->LECTURA CARACTER
8300 IF ASC(CAR$)<>27 THEN GOTO 8380
8310 YPOS=6:XPOS=13
8320 GOSUB 15800:REM -->POSICIONA EL CURSOR
```



```
8330 PRINT "
     "
8340 SW=3
8350 XPOS=13:YPOS=6
8360 GOSUB 1000:REM --->TECLADO
8370 LSET IN$=ALFA$
8380 REM
8390 YPOS=8:XPOS=11
8400 GOSUB 15800:REM -->POSICIONA EL CURSOR
8410 GOSUB 15900:REM -->LECTURA CARACTER
8420 IF ASC(CAR$)<>27 THEN GOTO 8500
8430 YPOS=8:XPOS=13
8440 GOSUB 15800:REM -->POSICIONA EL CURSOR
8450 PRINT "
     "
8460 SW=3
8470 GOSUB 15800:REM -->POSICIONA EL CURSOR
8480 GOSUB 1000:REM --->TECLADO
8490 LSET CI$=ALFA$
8500 REM
8510 YPOS=10:XPOS=11
8520 GOSUB 15800:REM -->POSICIONA EL CURSOR
8530 GOSUB 15900:REM -->LECTURA CARACTER
8540 IF ASC(CAR$)<>27 THEN GOTO 8620
8550 YPOS=10:XPOS=13
8560 GOSUB 15800:REM -->POSICIONA EL CURSOR
8570 PRINT "
     "
8580 SW=2
8590 XPOS=13:YPOS=10
8600 GOSUB 1000:REM --->TECLADO
8610 LSET TE$=MKS$(GRANDE#)
8620 REM
8630 YPOS=12:XPOS=11
8640 GOSUB 15800:REM -->POSICIONA EL CURSOR
8650 GOSUB 15900:REM -->LECTURA CARACTER
8660 IF ASC(CAR$)<>27 THEN GOTO 8740
8670 YPOS=12:XPOS=13
8680 GOSUB 15800:REM -->POSICIONA EL CURSOR
8690 PRINT "
     "
```

```
8700 SW=3
8710 YPOS=12:XPOS=13
8720 GOSUB 1000:REM --->TECLADO
8730 LSET PR$=ALFA$
8740 REM
8750 GOSUB 15500:REM -->ESCRITURA CLIENTE
8760 RETURN
8770 REM
9000 REM ::::::::::::::
9010 REM ::  BORRADO  ::
9020 REM ::::::::::::::
9030 REM
9040 GOSUB 500:REM ---->BORRADO PANTALLA
9050 YPOS=20:XPOS=1
9060 GOSUB 15800:REM -->POSICIONA EL CURSOR
9070 INPUT "CLAVE DE BUSQUEDA:";NOMINATIVO$
9080 IF LEN(NOMINATIVO$)=31 THEN GOTO 9100
9090 NOMINATIVO$=NOMINATIVO$+" ":GOTO 9080
9100 REM
9110 GOSUB 13000:REM REM -->CLAVE
9120 IF FIN=1 THEN GOTO 9250
9130 GOSUB 10000:REM -->MASCARA
9140 GOSUB 11000:REM -->VISUALIZACION
9150 XPOS=1:YPOS=20
9160 GOSUB 15800:REM -->POSICIONA EL CURSOR
9170 PRINT "¿QUIERE BORRARLO?(S/N)"
9180 GOSUB 15900:REM -->LECTURA CARACTER
9190 IF CAR$="S" OR CAR$="s" THEN GOTO 9210
9200 RETURN
9210 REM
9220 LSET TI$="C"
9230 GOSUB 15500:REM -->ESCRITURA CLIENTE
9240 RETURN
9250 REM
9260 YPOS=20:XPOS=1
9270 INPUT "NO EXISTE ESTE CLIENTE.PULSE <RETURN>";
     RESPUESTA$
9280 RETURN
10000 REM ::::::::::::::
10010 REM ::  MASCARA  ::
10020 REM ::::::::::::::
```



```
10030 REM
10040 YPOS=4:XPOS=1
10050 GOSUB 15800:REM -->POSICIONA EL CURSOR
10060 PRINT "CLIENTE     :"
10070 YPOS=6:XPOS=1
10080 GOSUB 15800:REM -->POSICIONA EL CURSOR
10090 PRINT "DIRECCION   :"
10100 YPOS=8:XPOS=1
10110 GOSUB 15800:REM -->POSICIONA EL CURSOR
10120 PRINT "CIUDAD      :"
10130 YPOS=10:XPOS=1
10140 GOSUB 15800:REM -->POSICIONA EL CURSOR
10150 PRINT "TELEFONO    :"
10160 YPOS=12:XPOS=1
10170 GOSUB 15800:REM -->POSICIONA EL CURSOR
10180 PRINT "PREFIJO     :"
10190 YPOS=14:XPOS=1
10200 GOSUB 15800:REM -->POSICIONA EL CURSOR
10210 PRINT "IZQUIERDO   :"
10220 YPOS=14:XPOS=20
10230 GOSUB 15800:REM -->POSICIONA EL CURSOR
10240 PRINT "DERECHO     :"
10250 RETURN
10260 REM
11000 REM ::::::::::::::::::::
11010 REM ::  VISUALIZACION  ::
11020 REM ::::::::::::::::::::
11030 REM
11040 YPOS=4:XPOS=13
11050 GOSUB 15800:REM -->POSICIONA EL CURSOR
11060 PRINT CL$;
11070 IF TI$<>" " THEN PRINT " BORRADO";
11080 YPOS=6:XPOS=13
11090 GOSUB 15800:REM -->POSICIONA EL CURSOR
11100 PRINT IN$
11110 YPOS=8:XPOS=13
11120 GOSUB 15800:REM -->POSICIONA EL CURSOR
11130 PRINT CI$
11140 YPOS=10:XPOS=13
11150 GOSUB 15800:REM -->POSICIONA EL CURSOR
11160 TELEFONO#=CVS(TE$)
```

```
11170 PRINT TELEFONO#
11180 YPOS=12:XPOS=13
11190 GOSUB 15800:REM -->POSICIONA EL CURSOR
11200 PRINT PR$
11210 YPOS=14:XPOS=13
11220 GOSUB 15800:REM -->POSICIONA EL CURSOR
11230 IZQUIERDO%=CVI(SI$)
11240 PRINT IZQUIERDO$
11250 YPOS=14:XPOS=33
11260 GOSUB 15800:REM -->POSICIONA EL CURSOR
11270 DERECHO%=CVI(DE$)
11280 PRINT DERECHO%
11290 RETURN
11300 REM
11500 REM ::::::::::::
11510 REM ::  LISTA  ::
11520 REM ::::::::::::
11530 REM
11540 FIN=0
11550 IF INICIO%>=ACTUAL% OR INICIO%<=0 THEN GOTO 11610
11560 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,4 AS
      TE$,4 AS PR$,1 AS TI$,2 AS SI$,2 AS DE$
11570 NUMERO%=INICIO%
11580 GOSUB 15000:REM -->LECTURA CLIENTE
11590 INICIO%=INICIO%+1
11600 RETURN
11610 FIN=1
11620 RETURN
11800 REM ::::::::::::::::
11810 REM :: VE CLIENTE  ::
11820 REM ::::::::::::::::
11830 REM
11840 FOR NUMERO%=2 TO ACTUAL%-1
11850 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,4 AS
      TE$,4 AS PR$,1 AS TI$,2 AS SI$,2 AS DE$
11860 GOSUB 15000:REM -->LECTURA CLIENTE
11870 GOSUB 500:REM ---->BORRADO PANTALLA
11880 GOSUB 10000:REM -->MASCARA
11890 GOSUB 11000:REM -->VISUALIZACION
11900 INPUT "PULSE <RETURN> PARA CONTINUAR";
      RESPUESTA$
```



```
11910 NEXT NUMERO%
11920 RETURN
12000 REM ::::::::::::::
12010 REM :: ACT NUDO  ::
12020 REM ::::::::::::::
12030 REM
12040 HECHO=0
12050 IF ACTUAL%=2 THEN LSET SI$=MKI$(0):LSET DE$=
      MKI$(0):GOTO 12520
12060 P%=2
12070 REM
12080 NUMERO%=P%
12090 GOSUB 15000:REM -->LECTURA CLIENTE
12100 IF NOMINATIVO$=CL$ THEN GOTO 12300
12110 IF CVI(SI$)<>0 THEN P%=CVI(SI$):GOTO 12500
12120 HECHO=1
12130 LSET SI$=MKI$(ACTUAL%)
12140 GOSUB 15500:REM -->ESCRITURA CLIENTE
12145 NUMERO%=ACTUAL%
12146 GOSUB 15000:REM -->LECTURA CLIENTE
12150 LSET SI$=MKI$(0)
12160 LSET DE$=MKI$(-P%)
12170 GOTO 12500
12300 REM
12310 IF CVI(DE$)>0 THEN P%=CVI(DE$):GOTO 12500
12320 HECHO=1
12330 M%=CVI(DE$)
12340 LSET DE$=MKI$(ACTUAL%)
12350 GOSUB 15500:REM -->ESCRITURA CLIENTE
12355 NUMERO%=ACTUAL%
12356 GOSUB 15000:REM -->LECTURA CLIENTE
12360 LSET DE$=MKI$(M%)
12370 LSET SI$=MKI$(0)
12500 REM
12510 IF HECHO=0 THEN GOTO 12080
12520 REM
12530 RETURN
12540 REM
13000 REM ::::::::::::
13010 REM ::  CLAVE  ::
13020 REM ::::::::::::
```

```
13030 REM
13040 ENC=0
13050 FIN=0
13060 ESTO%=2
13070 NUMERO%=2
13075 IF LEN(NOMINATIVO$)>=31 THEN GOTO 13078
13077 NOMINATIVO$=NOMINATIVO$+" ":GOTO 13075
13078 FIELD 1,31 AS CL$,33 AS IN$,15 AS CI$,4 AS
      TE$,4 AS PR$,1 AS TI$,2 AS SI$,2 AS DE$
13080 IF NUMERO%=0 THEN GOTO 13200
13090 GOSUB 15000:REM -->LECTURA CLIENTE
13100 ESTO%=NUMERO%
13110 IF NOMINATIVO$<CL$ THEN NUMERO%=CVI(SI$):
      GOTO 13080
13120 IF NOMINATIVO$=CL$ THEN ENC=1:PROXIMO%=CVI(
      DE$):GOTO 13300
13130 IF CVI(DE$)>0 THEN NUMERO%=CVI(DE$):GOTO
      13080
13140 NUMERO%=0
13200 FIN=1
13210 PROXIMO%=ESTO%
13220 RETURN
13300 IF FIN=1 OR ENC=1 THEN RETURN
13310 GOSUB 15000:REM -->LECTURA CLIENTE
3320 RETURN
13330 REM
13500 REM ::::::::::::::::
13510 REM ::  BUSQUEDA  ::
13520 REM ::::::::::::::::
13530 REM
13540 FUERA=0
13550 REM
13560 ESTO%=ABS(PROXIMO%)
13565 NUMERO%=ESTO%
13570 GOSUB 15000:REM -->LECTURA CLIENTE
13580 IF CVI(SI$)<>0 AND PROXIMO%>=0 THEN PROXIMO%=
      CVI(SI$):GOTO 13800
13590 FUERA=1
13600 PROXIMO%=CVI(DE$)
13800 IF FUERA=0 THEN GOTO 13550
13810 RETURN
```



```
13820 REM
15000 REM ::::::::::::::::::::::::::::::::
15010 REM ::  LECTURA FICHERO CLIENTE  ::
15020 REM ::::::::::::::::::::::::::::::::
15030 REM
15040 GET #1,NUMERO%
15050 RETURN
15090 REM
15500 REM ::::::::::::::::::::::::::::::::::
15510 REM ::  ESCRITURA FICHERO CLIENTE  ::
15520 REM ::::::::::::::::::::::::::::::::::
15530 REM
15550 PUT #1,NUMERO%
15560 RETURN
15590 REM
15800 REM ::::::::::::::::::::::::::
15810 REM ::  POSICIONA EL CURSOR  ::
15820 REM ::::::::::::::::::::::::::
15830 REM
15840 LOCATE YPOS,XPOS,1
15850 RETURN
15860 REM
15900 REM ::::::::::::::::::::::
15910 REM :: LECTURA CARACTER  ::
15920 REM ::::::::::::::::::::::
15930 REM
15940 CAR$=INPUT$(1)
15950 RETURN
15960 REM
```

AL_COMPACTA

```
100 REM ****************
110 REM *  AL COMPACTA  *
120 REM ****************
130 REM
140 OPEN "R",2,"CLIENTES.DAT",92
160 OPEN "R",1,"CLIENTE.VEC",92
170 FIELD 2,2 AS ACTUAL$,2 AS ULTIMO$
175 GET #2,1
```

```
180 FIELD 1,2 AS ACT$,2 AS ULT$
190 GET #1,1
210 X%=CVI(ULTIMO$)
220 Y%=CVI(ACT$)
230 NUMERO%=1
235 GOSUB 15500:REM -->ESCRITURA CLIENTE
240 ACTUAL%=2
250 Z%=2
260 REM
270 FIELD 1,31 AS CLI$,33 AS IND$,15 AS CIT$,4 AS
    TEL$,4 AS PRE$,1 AS TIP$,2 AS SIS$,2 AS DES$
280 GET #1,Z%
290 IF TIP$<>"C" THEN GOTO 320
300 PRINT CLI$;" BORRADO"
310 GOTO 475
320 REM
330 IF Z%>X% THEN GOTO 500
340 NOMINATIVO%=CLI$
345 FIELD 2,31 AS CL$,33 AS IN$,15 AS CI$,4 AS
    TE$,4 AS PR$,1 AS TI$,2 AS SI$,2 AS DE$
350 GOSUB 12000:REM -->ACT NUDO
360 LSET CL$=CLI$
370 LSET IN$=IND$
380 LSET CI$=CIT$
390 LSET TE$=TEL$
400 LSET PR$=PRE$
410 LSET TI$=TIP$
450 PRINT CL$
460 NUMERO%=ACTUAL%
465 GOSUB 15500:REM -->ESCRITURA CLIENTE
470 ACTUAL%=ACTUAL%+1
475 Z%=Z%+1
480 IF Z%<Y% THEN GOTO 260
490 GOTO 600
500 REM
510 PRINT "HAY ";Z%;" REGISTROS ACTIVOS EN UN
    FICHERO QUE LOS PREVEE ";X%;"!"
520 PRINT "NO PUEDO SEGUIR"
530 INPUT "PULSE <RETURN>:";RESPUESTA$
540 GOTO 650
600 REM
```



```
610 FIELD 2,2 AS ACT$,2 AS ULT$
620 NUMERO%=1
625 GOSUB 15000:REM -->LECTURA CLIENTE
630 LSET ACT$=MKI$(ACTUAL%)
640 GOSUB 15500:REM -->ESCRITURA CLIENTE
650 CLOSE 1
660 CLOSE 2
670 END
12000 REM ::::::::::::::::
12010 REM ::  ACT NUDO  ::
12020 REM ::::::::::::::::
12030 REM
12040 HECHO=0
12050 IF ACTUAL%=2 THEN LSET SI$=MKI$(0):LSET DE$=
      MKI$(0):GOTO 12520
12060 P%=2
12070 REM
12080 NUMERO%=P%
12090 GOSUB 15000:REM -->LECTURA CLIENTE
12100 IF NOMINATIVO$>CL$ THEN GOTO 12300
12110 IF CVI(SI$)<>0 THEN P%=CVI(SI$):GOTO 12500
12120 HECHO=1
12130 LSET SI$=MKI$(ACTUAL%)
12140 GOSUB 15500:REM -->ESCRITURA CLIENTE
12145 NUMERO%=ACTUAL%
12146 GOSUB 15000:REM -->LECTURA CLIENTE
12150 LSET SI$=MKI$(0)
12160 LSET DE$=MKI$(-P%)
12170 GOTO 12500
12180 REM
12300 REM
12310 IF CVI(DE$)>0 THEN P%=CVI(DE$):GOTO 12500
12320 HECHO=1
12330 M%=CVI(DE%)
12340 LSET DE$=MKI$(ACTUAL%)
12350 GOSUB 15500:REM -->ESCRITURA CLIENTE
12355 NUMERO%=ACTUAL%
12356 GOSUB 15000:REM -->LECTURA CLIENTE
12360 LSET DE$=MKI$(M%)
12370 LSET SI$=MKI$(0)
12500 REM
```

```
12510 IF HECHO=0 THEN 12080
12520 REM
12530 RETURN
12540 REM
15000 REM ::::::::::::::::::::::::::::::
15010 REM ::  LECTURA FICHERO CLIENTE  ::
15020 REM ::::::::::::::::::::::::::::::
15030 REM
15040 GET #2,NUMERO%
15050 RETURN
15060 REM
15500 REM ::::::::::::::::::::::::::::::::
15510 REM ::  ESCRITURA FICHERO CLIENTE  ::
15520 REM ::::::::::::::::::::::::::::::::
15530 REM
```


APENDICE B

APENDICE PASCAL

IS_PREPARA

```
PROGRAM is_prepara;
TYPE
    cli=RECORD
        CASE BOOLEAN OF
                TRUE:(cliente    :STRING[31];
                      direccion  :STRING[33];
                      ciudad     :STRING[15];
                      telefono   :INTEGER[8];
                      prefijo    :STRING[4];
                      tipo       :CHAR);
               FALSE:(primero    :INTEGER;
                      maximo     :INTEGER);

        END;
    ind=RECORD
        nombre     :STRING[31];
        izquierdo  :INTEGER;
        derecho    :INTEGER;
        END;
VAR
        cliente :FILE OF cli;
        indice  :FILE OF ind;
        x       :INTEGER;
        ultimo  :INTEGER;
```

```
BEGIN
 REWRITE(cliente,'cliente.dat');
 WRITE('¿de cuantos ficheros estara compuesto el fichero?:');
 READLN(ultimo);
 WITH cliente^
 DO BEGIN
      primero:=2;
      maximo:=ultimo+1;
      PUT(cliente);
      PUT(cliente);
      cliente:='';
      direccion:='';
      ciudad:='';
      telefono:=0;
      prefijo:='0000';
      tipo:=' ';
      FOR X:=1 TO ultimo DO PUT(cliente);
 END;
 CLOSE
 REWRITE
 WITH indice^
 DO BEGIN
      nombre:='';
      izquierdo:=0;
      derecho:=0;
      FOR x:=0 TO ultimo+1 DO PUT(indice);
 END;
 CLOSE(indice.LOCK);
END.
```

IS_GESTION

```
PROGRAM is_gestion;
TYPE
    cli=RECORD
        CASE BOOLEAN OF
                TRUE:(cliente    :STRING[31];
                      direccion  :STRING[33];
                      ciudad     :STRING[15];
                      telefono   :INTEGER[8];
```



```
                      prefijo    :STRING[4];
                      tipo       :CHAR);
               FALSE:(primero    :INTEGER;
                      maximo     :INTEGER);

        END;

    ind=RECORD
        nombre     :STRING[31];
        izquierdo  :INTEGER;
        derecho    :INTEGER;
        END;
TYPE
        cliente:FILE OF cli;
        indice:FILE OF ind;
        hecho,
        fuera,fin:BOOLEAN;
        partida,termino,nominativo:STRING[33];
        car,sw,respuesta,que_quiere:CHAR;
        st:STRING[1];
        grande:INTEGER[15];
        x_pos,y_pos,p,m,x,y,z,j,inicio,alt,pequeno,
        cual,presente,esto,proximo,actual,
        ultimo:INTEGER:

PROCEDURE borrado_pantalla;
BEGIN
  WRITE(CHR$(28));
END;

PROCEDURE teclado;
BEGIN
 CASE sw OF
        '1':pequeno:=0;
        '2':grande:=0;
        '3':alfa:='';
   END;
   GOTOXY(x_pos,y_pos);
   REPEAT
    READ(KEYBOARD,car);
    IF NOT EOLN(KEYBOARD)
```

```
THEN BEGIN
 IF ORD(car)=8
  THEN BEGIN
    GOTOXY(x_pos,y_pos);


    CASE sw OF
     '1':BEGIN
            PEQUENO:=PEQUENO div 10;
            WRITE(PEQUENO,' ');
            GOTOXY(x_pos,y_pos);
            WRITE(pequeno);
         END;
     '2':BEGIN
            grande:=grande DIV 10;
            WRITE(grande,' ');
            GOTO(x_pos,y_pos);
           WRITE(grande);
          END;
     '3':BEGIN
               DELETE(alfa,LEGTH(alfa),1);
               WRITE(alfa,' ');
               GOTOXY(x_pos,y_pos);
               WRITE(alfa);
         END;
     END;
   END
   ELSE BEGIN
    IF sw IN['1','2']
    THEN BEGIN
      IF car
      THEN pequeno:=pequeno*10+ORD(car)-48
      ELSE grande:=grande*10+ORD(car)-48;
      WRITE(car);
    END
    ELSE WRITE(CHR$(7));
   END  {del campo numerico}
   ELSE BEGIN
    IF car IN[' '..'"']
    THEN BEGIN
     st:=' ';
```



```
       st[1]:=car;
       alfa:=CONCAT(alfa,st);
       WRITE(car);
      END
      ELSE WRITE(CHR$(7));
    END; {del campo alfabetico}
   END;  {si no tecla de retorno}
  END;   {no es return}
 UNTIL EOLN(KEYBOARD);
 END;

 PROCEDURE menu;
 BEGIN
  borrado_pantalla;
     WRITELN;
     WRITELN('1 --> carga de datos');
     WRITELN;
     WRITELN('2 --> lectura con el numero de registro');
     WRITELN;
     WRITELN('3 --> lectura con el nombre de cliente');
     WRITELN;
     WRITELN('4 --> modificacion de un registro');
     WRITELN;
     WRITELN('5 --> borrado de un registro');
     WRITELN;
     WRITELN('9 --> fin del programa');
     WRITELN;
     READ(que_quiere);WRITELN;
    END;

  PROCEDURE visualizacion;
  BEGIN
   WITH cliente^
   DO BEGIN
        GOTOXY(13,4);
        WRITE(cliente^.cliente);
        IF tipo<>' ' THEN WRITE(' borrado')
                     ELSE WRITE('        ');
       GOTOXY(13,6);
       WRITE(direccion);
       GOTOXY(13,8);
```

```
      WRITE(ciudad);
      GOTOXY(13,10);
      WRITE(telefono);
      GOTOXY(13,12);
      WRITE(prefijo);
  END;
END;

PROCEDURE mascara;
BEGIN
      GOTOXY(0,4);
      WRITE('cliente      :');
      GOTOXY(0,6);
      WRITE('direccion    :');
      GOTOXY(0,8);
      WRITE('ciudad       :');
      GOTOXY(0,10);
      WRITE('telefono     :');
      GOTOXY(0,12);
      WRITE('prefijo      :');
END;

PROCEDURE clave;
BEGIN
  fuera:=FALSE;
  fin:=FALSE;
  esto:=2;
  j:=2;
  WITH indice^
  DO BEGIN
   REPEAT
    IF j=0
    THEN BEGIN
      fin:=TRUE;
      proximo:=esto;
    END;
    ELSE BEGIN
    SEEK(indice,j);
    GET(indice);
    IF nominativo<nombre
     THEN j:=izquierdo
```



```
    ELSE IF nominativo=nombre
         THEN BEGIN
              fuera:=TRUE;
              proximo:=derecho;
            END
            ELSE IF derecho>0 THEN j:=derecho
                                 ELSE j:=0;
          END;
        UNTIL fuera OR fin;
    END;
  IF NOT fin
   THEN BEGIN
     SEEK(cliente,j);
     GET(cliente);
   END;
 END;


PROCEDURE modificacion;
BEGIN
  borrado_pantalla;
  GOTOXY(0,20);
  WRITE('clave a buscar: ');
  sw:='3';
  x_pos:=23;
  y_pos:=20;
  teclado;
  nominativo:=alfa;
  clave;
  IF NOT fin
  THEN BEGIN
    mascara;
    visualizacion;
    GOTOXY(0,20);
    WRITE('pulse <ESC> para modificar el campo,',
          '<RETURN> para continuar');


    GOTOXY(13,6);
    READ(respuesta);
    IF respuesta=CHR$(27)
```

```
 THEN BEGIN
   GOTOXY(13,6);
   WRITE('                              ');
   sw:='3';
   x_pos:=13;
   y_pos:=6;
   teclado;
   cliente^.direccion:=alfa;
 END;

 GOTOXY(13,8);
 READ(respuesta);
 IF respuesta=CHR$(27)
 THEN BEGIN
   GOTOXY(13,8);
   WRITE('                    ');
   sw:='3';
   x_pos:=13;
   y_pos:=8;
   teclado;
   cliente^.ciudad:=alfa;
 END;

 GOTOXY(13,10);
 READ(respuesta);
 IF respuesta=CHR$(27)
 THEN BEGIN
   GOTOXY(13,10);
   WRITE('                          ');
   sw:='2';
   x_pos:=13;
   y_pos:=10;
   teclado;
   cliente^.telefono:=grande;
 END;

 GOTOXY(13,12);
 READ(respuesta);
 IF respuesta=CHR$(27)
 THEN BEGIN
   GOTOXY(13,12);
```



```
      WRITE('                    ');
      sw:='3';
      x_pos:=13;
      y_pos:=12;
      teclado;
      cliente^.prefijo:=alfa;
    END;
  SEEK(cliente,esto);
  PUT(cliente);
END

ELSE BEGIN
   GOTOXY(0,20);
   WRITE('no existe este registro,pulse <RETURN>');
   READLN;
  END;
END;

PROCEDURE borrado;
BEGIN
   borrado_pantalla;
   GOTOXY(0,20);
   WRITE('clave a buscar: ');
   sw:='3';
   x_pos:=23;
   y_pos:=20;
   teclado;
   nominativo:=alfa;
   clave;
   IF NOT fin
   THEN BEGIN
     mascara;
     visualizazion;
     GOTOXY(0,20);
     WRITE('¿quiere borrarlo?(S/N)');
     READ(respuesta);
     IF respuesta IN ['S','s']
     THEN BEGIN
       cliente^.tipo:='C'
       SEEK(cliente,esto);
       PUT(cliente);
```

```
          END;
        END
        ELSE BEGIN
         GOTOXY(0,20);
         WRITE('no existe este registro,pulse <RETURN>');
         READLN;
        END;

  END;
 PROCEDURE busqueda;
 BEGIN
   WITH indice^
   DO BEGIN
       fuera:=FALSE;
       REPEAT
         presente:=ABS(cual);
         SEEK(indice,presente);
         GET(indice);
         SEEK(cliente,presente);
         GET(cliente);
         IF(izquierdo<>0)
         AND (cual>=0)
         THEN cual:=izquierdo
         ELSE BEGIN
          fuera:=TRUE;
          cual:=derecho;
         END;
      UNTIL fuera;
    END;  {WITH indice^}
 END;

 PROCEDURE para_nombre;
 BEGIN
   GOTOXY(0,20);
   WRITE('nombre del primer cliente  :');
   sw:='3';
   x_pos:=32;
   y_pos:=20;
   teclado;
   partida:=alfa;
   GOTOXY(0,20);
```



```
  WRITE('nombre del ultimo cliente  :',
       '                            ');
  sw:='3';
  x_pos:=32;
  y_pos:=20;
  teclado;
  termino:=alfa;
  nominativo:=partida;
  clave;
  cual:=proximo;
  IF fin
  THEN BEGIN
   busqueda;
   fin:=FALSE;
  END;
  REPEAT
   borrado_pantalla;
   mascara;
   visualizacion;
   GOTOXY(0,20);
   busqueda;
   WRITE('<ESPACIO> para continuar,',
         '<ESC> para terminar');
   READ(respuesta);
  UNTIL(respuesta=CHR$(27))
     OR(presente=0)
     OR(indice^.nombre>termino)
 END;

PROCEDURE act_nudo;
BEGIN
  WITH indice^
  DO BEGIN
        hecho:=FALSE;
        IF actual=2
        THEN BEGIN   {si primer registro}
         nombre:=nominativo;
         izquierdo:=0;
         derecho:=0;
         SEEK(indice,2);
         PUT(indice);
```

```
END
ELSE BEGIN
   p:=2;
   REPEAT
      SEEK(indice,p);
      GET(indice);
      IF nominativo>cliente
      THEN BEGIN
        IF derecho>0
          THEN p:=derecho
          ELSE BEGIN
              hecho:=TRUE;
              m:=derecho;
              derecho:=actual;
              SEEK(indice,p);
              PUT(indice);
              SEEK(indice,actual);
              derecho:=m;
              izquierdo:=0;
              nombre:=nominativo;
              PUT(indice);
           END;
        END
        ELSE BEGIN
        IF izquierdo<>0
          THEN p:=izquierdo;
          ELSE BEGIN
              hecho:=TRUE;
              izquierdo:=actual;
              SEEK(indice,p);
              PUT(indice);
              SEEK(indice,actual);
              izquierdo:=0;
              derecho:=-p;
              nombre:=nominativo;
              PUT(indice);
           END;
        END;
     UNTIL hecho;
   END;
   actual:=actual+1;
```



```
     END;
  END;

  PROCEDURE lista;
  BEGIN
    fin:=FALSE;
    IF (inicio<actual)
       AND (inicio>0)
       THEN BEGIN
          SEEK(cliente,inicio);
          GET(cliente);
          inicio:=SUCC(inicio);
       END
       ELSE fin:=TRUE;
    END;

  PROCEDURE lectura;
   BEGIN
    GOTOXY(0,20);
    WRITE('numero del primer registro: ');
    sw:='1';
    x_pos:=32;
    y_pos:=20;
    teclado;
    inicio:=pequeno+1
    GOTOXY(0,20);
    WRITE('numero del ultimo registro: ');
    sw:='1';
    x_pos:=32;
    y_pos:=20;
    teclado;
    alt:=pequeno+1;
    IF alt>actual-1 THEN alt:=actual-1;
    lista;
    IF NOT fin
    THEN REPEAT
      borrado_pantalla;
      mascara;
      visualizacion;
      GOTOXY(0,20);
      lista;
```

```
     WRITE('<ESPACIO> para continuar,',
           '<ESC> para terminar');
     READ(respuesta);
    UNTIL  (respuesta=CHR$(27))
        OR  fin
        OR (inicio>alt+1);
END;

PROCEDURE carga;
BEGIN
  borrado_pantalla;
  WRITELN('la carga parte con el numero',
          actual-1);
  IF actual>2
  THEN BEGIN
        SEEK(cliente,PRED(actual));
        GET(cliente);
        WRITELN('el registro anterior escrito',
                'es el siguiente:');
        mascara;
        visualizacion;
        WRITELN;
    END;

WRITELN('pulse <RETURN> para iniciar la carga');
READLN;
REPEAT
  borrado_pantalla;
GOTOXY(0,20);
WRITELN('para terminar,responda "zzzz" a la',
       'peticion del nombre del cliente');
mascara;
sw:='3';
x_pos:=13;
y_pos:=4;
nominativo:=alfa;
IF nominativo<>'zzzz'
 THEN BEGIN
   IF actual>ultimo
   THEN BEGIN
        WRITE('no puedo seguir.se ha superado',
```



```
              'la dimension maxima del fichero.',
              'pulse <RETURN>');
        READLN;
        EXIT(carga);
      END;
     clave;
     IF NOT fin
      THEN BEGIN
       GOTOXY(0,20);
       WRITE('este cliente ya existe.no puedo',
             'insertarlo.pulse <RETURN>');
       READLN;
       EXIT(carga);
      END;
    WITH cliente^
    DO BEGIN
        cliente:=nominativo;
        sw:='3';
        x_pos:=13;
        y_pos:=6;
        teclado;
        direccion:=alfa;
        sw:='3';
        x_pos:=13;
        y_pos:=8;
        teclado;
        ciudad:=alfa;
        sw:='2';

        x_pos:=13;
        y_pos:=10;
        teclado;
        telefono:=grande;
        sw:='3';
        x_pos:=13;
        y_pos:=12;
        teclado;
        prefijo:=alfa;
        tipo:=' ';
        SEEK(cliente,actual);
        PUT(cliente);
```

```
        act_nudo;
        primero:=actual;
        maximo:=ultimo;
        SEEK(cliente,1);
        PUT(cliente);
        END;  {de la with cliente}
      END;    {si cliente}
    UNTIL nominativo='zzzz';
  END;

 BEGIN
 RESET(indice,'indice.dat');
 RESET(cliente,'cliente.dat');
 SEEK(cliente,1);
 GET(cliente);
 actual:=cliente^.primero;
 ultimo:=cliente^.maximo;
 REPEAT
   menu;
   CASE que_quiere OF
        '1':carga;
        '2':lectura;
        '3':para_nombre;
        '4':modificacion;
        '5':borrado;
      END;
   UNTIL que_quiere='9';
   CLOSE (indice.LOCK);
   CLOSE(cliente.LOCK);
END.
```

IS_COMPACTA

```
PROGRAM is_compacta;
TYPE
     cli=RECORD
        CASE BOOLEAN OF
          TRUE:(cliente:STRING[31];
                direccion:STRING[33];
                ciudad   :STRING[15];
```



```
                telefono :INTEGER[8];
                prefijo  :STRING[4];
                tipo     :CHAR);
          FALSE:(primero  :INTEGER;
                 maximo   :INTEGER);
   END;

    ind=RECORD
       nombre    :STRING[31];
       izquierdo :INTEGER;
       derecho   :INTEGER;
       END;

VAR viejo,cliente:FILE OF cli;
    indice:FILE OF ind;
    nominativo:STRING[31];
    hecho:BOOLEAN;
    m,p,x,y,z,actual,ultimo:INTEGER;

PROCEDURE act_nudo;
BEGIN
  WITH indice^
  DO BEGIN
     hecho:=FALSE;
     IF actual=2
     THEN BEGIN  {si primer registro}
        nombre:=nominativo;
        izquierdo:=0;
        derecho:=0;
        SEEK(indice,2);
        PUT(indice);
     END
     ELSE BEGIN
      p:=2;
      REPEAT
        SEEK(indice,p);
        GET(indice);
        IF nominativo>nombre
        THEN BEGIN
          IF derecho>0
          THEN p:=derecho;
```

```
        ELSE BEGIN
         hecho:=TRUE;
         m:=derecho;
         derecho:=actual;
         SEEK(indice,p);
         PUT(indice);
         SEEK(indice,actual);
         derecho:=m;
         izquierdo:=0;
         nombre:=nominativo;
         PUT(indice);
          END;
         END;
         ELSE BEGIN
              IF IZQUIERDO<>0
              THEN p:=izquierdo
              ELSE BEGIN
              hecho:=TRUE;
              izquierdo:=actual;
              SEEK(indice,p);
              PUT(indice);
              SEEK(indice,actual);
              izquierdo:=0;
              derecho:=-p;
              nombre:=nominativo;
              PUT(indice)
           END;
        END;
      UNTIL hecho;
     END;
     actual:=actual+1;
   END;   {WITH indice^}
END;

BEGIN
 RESET(viejo,'cliente.vie');
 RESET(indice,'indice.dat');
 RESET(cliente,'cliente.dat');
 SEEK(cliente,1);
 GET(cliente);
 SEEK(viejo,1);
```



```
GET(viejo);
ultimo:=cliente^.maximo;
y:=viejo^.primero;
GET(viejo);
actual:=2;
z:=2;
REPEAT
 WITH viejo^
 DO BEGIN
       IF tipo='C'
       THEN WRITELN(cliente:15,' borrado')
       ELSE BEGIN
         WRITELN(cliente:15);
         cliente^:=viejo^;
         SEEK(cliente,actual);
         PUT(cliente);
         nominativo:=viejo^.cliente;
         act_nudo;
       END;
       IF actual-1>ultimo
       THEN BEGIN
          WRITELN('hay',actual-1,' registros',
                 'activos en un fichero que los',
                 'prevee ',ultimo,'!');
          WRITELN('no puedo seguir.');
          WRITELN('pulse <RETURN>');
          READLN;
          EXIT(program);
        END;
        GET(viejo);
        z:=z+1;
      END;  {de la with viejo}
 UNTIL z>=y;
 SEEK(cliente,1);
 GET(cliente);
 cliente^.primo:=actual;
 SEEK(cliente,1);
 PUT(cliente);
 CLOSE(viejo);
 CLOSE(indice);
 CLOSE(cliente);
END.
```

HAS_PREPARA

```
PROGRAM hash_prepara;
CONST
    divisor=7;
TYPE
     cli=RECORD
        CASE BOOLEAN OF
          TRUE:(cliente  :STRING[31];
                direccion:STRING[33];
                ciudad   :STRING[15];
                telefono :INTEGER[8];
                prefijo  :STRING[4];
                tipo     :CHAR;
                colision :INTEGER);
         FALSE:(primero  :INTEGER;
                maximo   :INTEGER);
    END;
VAR
     cliente:FILE OF cli;
     indice :FILE OF INTEGER;
     x,y,z  :INTEGER;
     actual,ultimo:INTEGER;

BEGIN
  REWRITE(cliente,'cliente.dat');
  WRITE('¿de cuantos registros estara compuesto',
        'el fichero?');
  READLN(ultimo);

  WITH cliente^
  DO BEGIN
        primero:=2;
        maximo:=ultimo+1;
        PUT(cliente);
        PUT(cliente);
        cliente:='';
        direccion:='';
        ciudad:='';
        telefono:=0;
        prefijo:='0000';
```



```
        tipo:=' ';
        colision:=-1;
        FOR x:=2 TO ultimo+1 DO PUT(cliente);
  END;
  CLOSE(cliente.LOCK);
  REWRITE(indice,'indice.dat');
  indice^:=-1;
  FOR x:=0 TO divisor-1 DO PUT(indice);
  CLOSE(indice,LOCK);
END.
```

HAS_GESTION

```
PROGRAM hash_gestion;
CONST
    divisor=7;
TYPE
     cli=RECORD
        CASE BOOLEAN OF
          TRUE:(cliente  :STRING[31];
                direccion:STRING[33];
                ciudad   :STRING[15];
                telefono :INTEGER[8];
                prefijo  :STRING[4];
                tipo     :CHAR;
                colision :INTEGER);
         FALSE:(primero  :INTEGER;
                maximo   :INTEGER);
    END;

VAR
        cliente:FILE OF cli;
        indice:FILE OF INTEGER;
        fuera,fin:BOOLEAN;
        partida,termino,alfa,nominativo:STRING[33];
        car,sw,respuesta,que_quiere:CHAR;
        st:STRING[1];
        grande:INTEGER[15];
        j,x_pos,y_pos,cual,inicio,alt,
        pequeno,hash,actual,ultimo:INTEGER;
        x,y:INTEGER[8];
```

```
        r1,r2:INTEGER[8];
        z:INTEGER[8];
        c,n:INTEGER

PROCEDURE borrado_pantalla;
BEGIN
  WRITE(CHR$(28));
END;

PROCEDURE teclado;
BEGIN
  CASE sw OF
        '1':pequeno:=0;
        '2':grande:=0;
        '3':alfa:='';
END;
GOTOXY(x_pos,y_pos);
REPEAT
  READ(KEYBOARD,car);
  IF NOT EOLN(KEYBOARD)
  THEN BEGIN
     IF ORD(car)=8
      THEN BEGIN
      GOTOXY(x_pos,y_pos);
      CASE sw OF
        '1':BEGIN
              pequeno:=pequeno DIV 10;
              WRITE(pequeno,' ');
              GOTOXY(x_pos,y_pos);
      UNTIL EOLN(KEYBOARD);
     END;
              WRITE(pequeno);
            END;
        '2':BEGIN
             grande:=grande DIV 10;
             WRITE(grande,' ');
             GOTOXY(x_pos,y_pos);
             WRITE(grande);
           END;
        '3':BEGIN
             delete(alfa,LENGTH(alfa),1);
```



```
              WRITE(alfa,' ');
              GOTOXY(x_pos,y_pos);
              WRITE(alfa);
            END;
      END;
    END
    ELSE BEGIN
      IF sw IN ['1','2']
      THEN BEGIN
      IF car IN ['0'..'9']
         THEN BEGIN
           IF sw='1'
           THEN pequeno:=pequeno*10+ORD(car)-48
           ELSE grande:=grande*10+ORD(car)-48;
           WRITE(car);
         END
         ELSE WRITE(CHR$(7));
       END   {del campo numerico}
       ELSE BEGIN
         IF car IN [' '..'"']
         THEN BEGIN
           st:=' ';
           st[1]:=car;
           alfa:=CONCAT(alfa,st);
           WRITE(car);
         END
         ELSE WRITE(CHR$(7));
       END;  {del campo alfabetico}
     END;    {si no tecla de retorno}
   END;      {no es RETURN}

PROCEDURE menu;
BEGIN
 borrado_pantalla;
    WRITELN;
    WRITELN('1 --> carga de datos');
    WRITELN;
    WRITELN('2 --> lectura con el numero de registro');
    WRITELN;
    WRITELN('3 --> lectura con el nombre de cliente');
    WRITELN;
```

```
    WRITELN('4 --> modificacion de un registro');
    WRITELN;
    WRITELN('5 --> borrado de un registro');
    WRITELN;
    WRITELN('9 --> fin del programa');
    WRITELN;
    READ(que_quiere);WRITELN;
   END;

PROCEDURE visualizacion;
BEGIN
 WITH cliente^
 DO BEGIN
      GOTOXY(13,4);
      WRITE(cliente^.cliente);
      IF tipo<>' ' THEN WRITE(' borrado')
                   ELSE WRITE('        ');
     GOTOXY(13,6);
     WRITE(direccion);
     GOTOXY(13,8);
     WRITE(ciudad);
     GOTOXY(13,10);
     WRITE(telefono);
     GOTOXY(13,12);
     WRITE(prefijo);
     GOTOXY(13,14);
     WRITE(colision);
  END;
END;


PROCEDURE mascara;
BEGIN
      GOTOXY(0,4);
      WRITE('cliente       :');
      GOTOXY(0,6);
      WRITE('direccion     :');
      GOTOXY(0,8);
      WRITE('ciudad        :');
      GOTOXY(0,10);
      WRITE('telefono      :');
```



```
      GOTOXY(0,12);
      WRITE('prefijo       :');
      GOTOXY(0,14);
      WRITE('colision      :);
END;

PROCEDURE busqueda;
BEGIN
  fin:=FALSE;
  SEEK(indice,hash);
  GET(indice);
  cual:=indice^;
  REPEAT
   IF cual=-1
   THEN fin:=TRUE
   ELSE BEGIN
        SEEK(cliente,cual);
        GET(cliente);
        IF nominativo<>cliente^.cliente
        THEN cual:=cliente^.colision;
      END;

PROCEDURE calculo;
BEGIN
  x:=0;
  y:=0;
  FOR n:=1 TO LENGTH(nominativo)
  DO BEGIN
     c:=ORD(nominativo[n]);
     IF ODD(n) THEN x:=x+c
               ELSE y:y+c;
 END;
 r1:=x-((x DIV 256)*256);
 r2:=y-((y DIV 256)*256);
 z:=r2*256+r1
 hash:=TRUNC(z-((z DIV divisor)*divisor));
 IF hash=0 THEN hash:=divisor;
END;

PROCEDURE modificacion;
BEGIN
```

```
borrado_pantalla;
GOTOXY(0,20);
WRITE('clave a buscar: ');
sw:='3';
x_pos:=23;
y_pos:=20;
teclado;
nominativo:=alfa;
calculo;
busqueda;
IF NOT fin
THEN BEGIN
  mascara;
  visualizacion;
  GOTOXY(0,20);
  WRITE('pulse <ESC> para modificar el campo,',
        '<RETURN> para continuar');

  GOTOXY(13,6);
  READ(respuesta);
  IF respuesta=CHR$(27)
  THEN BEGIN
    GOTOXY(13,6);
    WRITE('                              ');
    sw:='3';
    x_pos:=13;
    y_pos:=6;
    teclado;
    cliente^.direccion:=alfa;
  END;

  GOTOXY(13,8);
  READ(respuesta);
  IF respuesta=CHR$(27)
  THEN BEGIN
    GOTOXY(13,8);
    WRITE('                 ');
    sw:='3';
    x_pos:=13;
    y_pos:=8;
    teclado;
```



```
      cliente^.ciudad:=alfa;
    END;

    GOTOXY(13,10);
    READ(respuesta);
    IF respuesta=CHR$(27)
    THEN BEGIN
      GOTOXY(13,10);
      WRITE('                    ');
      sw:='2';
      x_pos:=13;
      y_pos:=10;
      teclado;
      cliente^.telefono:=grande;
    END;

    GOTOXY(13,12);
    READ(respuesta);
    IF respuesta=CHR$(27)
    THEN BEGIN
      GOTOXY(13,12);
      WRITE('                    ');
      sw:='3';
      x_pos:=13;
      y_pos:=12;
      teclado;
      cliente^.prefijo:=alfa;
    END;
  SEEK(cliente,esto);
  PUT(cliente);
END

ELSE BEGIN
  GOTOXY(0,20);
  WRITE('no existe este registro,pulse <RETURN>');
  READLN;
 END;
END;

PROCEDURE borrado;
BEGIN
```

```
  borrado_pantalla
  GOTOXY(0,20);
  WRITE('clave a buscar');
  READLN(nominativo);
  calculo;
  busqueda;
  IF NOT fin
  THEN BEGIN
   mascara;
   visualizacion;
   GOTOXY(0,20);
   WRITE(¿quiere borrarlo?(S/N)');
   READ(respuesta);
   IF respuesta IN ['s','S']
     THEN BEGIN
     cliente^.tipo:='C';
     SEEK(cliente,cual);
     PUT(cliente);
    END;
 END
 ELSE BEGIN
  GOTOXY(0,20);
  WRITE('no existe este cliente,pulse');

 END;
END;

PROCEDURE nombre;
BEGIN
  GOTOXY(0,20);
  WRITE('nombre del cliente     :');
  sw:='3'
  x_pos:=32;
  y_pos:=20;
  teclado;
  nominativo:=alfa;
  calculo;
  busqueda;
  IF NOT fin
  THEN BEGIN
  borrado_pantalla;
```



```
    mascara;
    visualizacion;
    GOTOXY(0,20);
    WRITE('pulse <RETURN> para continuar');
    READLN;
  END;
  ELSE BEGIN
    GOTOXY(0,20
    WRITE('no existe este cliente,pulse <RETURN>');
    READLN;
  END;
 END;

PROCEDURE lista;
BEGIN
  fin:=FALSE;
  IF (inicio<actual)
     AND (inicio>0)
     THEN BEGIN
        SEEK(cliente,inicio);
        GET(cliente);
        inicio:=SUCC(inicio);
      END
      ELSE fin:=TRUE;
    END;

PROCEDURE lectura;
BEGIN
  GOTOXY(0,20);
  WRITE('numero del primer registro: ');
  sw:='1';
  x_pos:=32;
  y_pos:=20;
  teclado;
  inicio:=pequeno+1
  GOTOXY(0,20);
  WRITE('numero del ultimo registro: ');
  sw:='1';
  x_pos:=32;
  y_pos:=20;
  teclado;
```

```
  alt:=pequeno+1;
  IF alt>actual-1 THEN alt:=actual-1;
  lista;
  IF NOT fin
  THEN REPEAT
    borrado_pantalla;
    mascara;
    visualizacion;
    GOTOXY(0,20);
    lista;
    WRITE('<ESPACIO> para continuar,',
          '<ESC> para terminar');
    READ(respuesta);
   UNTIL  (respuesta=CHR$(27))
       OR  fin
       OR (inicio>alt+1);
END;

PROCEDURE carga;
BEGIN
  borrado_pantalla;
  WRITELN('la carga parte con el numero',
          actual-1);
  IF actual>2
  THEN BEGIN
        SEEK(cliente,PRED(actual));
        GET(cliente);
        WRITELN('el registro anterior escrito',
                'es el siguiente:');
        mascara;
        visualizacion;
        WRITELN;
    END;

    WRITELN('pulse <RETURN> para iniciar la carga');
    READLN;
    REPEAT
      borrado_pantalla;
    GOTOXY(0,20);
    WRITELN('para terminar,responda "zzzz" a la',
            'peticion del nombre del cliente');
```



```
mascara;
sw:='3';
x_pos:=13;
y_pos:=4;
teclado;
nominativo:=alfa;
IF nominativo<>'zzzz'
 THEN BEGIN
   IF actual>ultimo
   THEN BEGIN
        WRITE('no puedo seguir.se ha superado',
              'la dimension maxima del fichero.',
              'pulse <RETURN>');
        READLN;
        EXIT(carga);
      END;

calculo;
busqueda;
IF NOT fin
THEN BEGIN
  GOTOXY(0,20);
   WRITE('<ESPACIO> para seguir,<ESC> para terminar');
   READ(respuesta);
   WRITE('este cliente ya existe.no puedo',
         'insertarlo.pulse <RETURN>');
   READLN;
   EXIT(carga);
  END;
WITH cliente^
DO BEGIN
    cliente:=nominativo;
    sw:='3';
    x_pos:=13;
    y_pos:=6;
    teclado;
    direccion:=alfa;
    sw:='3';
    x_pos:=13;
    y_pos:=8;
    teclado;
```

```
        ciudad:=alfa;
        sw:='2';
        x_pos:=13;
        y_pos:=10;
        teclado;
        telefono:=grande;
        sw:='3';
        x_pos:=13;
        y_pos:=12;
        teclado;
        prefijo:=alfa;
        tipo:=' ';
        SEEK(indice,hash);
        GET(indice);
        colision:=indice^;
        indice^:=actual;
        SEEK(indice,hash);
        PUT(indice);
        SEEK(cliente,actual);
        PUT(cliente);
        actual:=SUCC(actual);
        primero:=actual;
        maximo:=ultimo;
        SEEK(cliente,1);
        PUT(cliente);
        END;  {de la with cliente}
      END;    {si cliente}
    UNTIL nominativo='zzzz';
  END;
BEGIN
 RESET(indice,'indice.dat');
 RESET(cliente,'cliente.dat');
 SEEK(cliente,1);
 GET(cliente);
 actual:=cliente^.primero;
 ultimo:=cliente^.maximo;
 REPEAT
   menu;
   CASE que_quiere OF
        '1':carga;
        '2':lectura;
```



```
        '3':nombre;
        '4':modificacion;
        '5':borrado;
      END;
   UNTIL que_quiere='9';
   CLOSE (indice.LOCK);
   CLOSE(cliente.LOCK);
END.
```

HAS_COMPACTA

```
PROGRAM has_compacta;
COST
  divisor=7;
TYPE
    cli=RECORD
        CASE BOOLEAN OF
                TRUE:(cliente    :STRING[31];
                      direccion  :STRING[33];
                      ciudad     :STRING[15];
                      telefono   :INTEGER[8];
                      prefijo    :STRING[4];
                      tipo       :CHAR;
                      colision   :INTEGER);
               FALSE:(primero    :INTEGER;
                      maximo     :INTEGER);

        END;
VAR
        viejo,cliente:FILE OF cli;
        indice:FILE OF INTEGER;
        nominativo:STRING[31];
        hecho:BOOLEAN;
        x,y,r1,r2,z:INTEGER[8];
        actual,hash,c,n,ultimo:INTEGER;

PROCEDURE calculo;
BEGIN
  x:=0;
  y:=0;
```

```
  FOR n:=1 TO LENGTH(nominativo)
  DO BEGIN
     c:=ORD(nominativo[n]);
     IF ODD(n) THEN x:=x+c
               ELSE y:y+c;
 END;
 r1:=x-((x DIV 256)*256);
 r2:=y-((y DIV 256)*256);
 z:=r2*256+r1
 hash:=TRUNC(z-((z DIV divisor)*divisor));
 IF hash=0 THEN hash:=divisor;
END;
BEGIN
 RESET(viejo,'cliente.vie');
 RESET(indice,'indice.dat');
 RESET(cliente,'cliente.dat');
 SEEK(viejo,1);
 GET(viejo);
 SEEK(cliente,1);
 GET(cliente);
 ultimo:=cliente^.maximo;
 i:=viejo^.primero;
 j:=2;
 GET(viejo);
 actual:=2;
 k:=2;
 REPEAT
  WITH viejo^
  DO BEGIN
       IF tipo='C'
       THEN WRITELN(cliente:15,' borrado')
       ELSE BEGIN
       k:=k+1;
         WRITELN(cliente:15);
         cliente^.cliente:=cliente;
         cliente^.direccion:=direccion;
         cliente^.ciudad:=ciudad;
         cliente^.telefono:=telefono;
         cliente^.prefijo:=prefijo;
         cliente^.tipo:=' ';
         nominativo:=viejo^.cliente;
```



```
             calculo;
             SEEK(indice,hash);
             GET(indice);
             cliente^.colision:=indice^;
             indice^:=actual;
            SEEK(indice,hash);
            PUT(indice);
            SEEK(cliente,actual);
            actual:=actual+1;
          END;
          j:=j+1;
          GET(viejo);
          IF actual-1>ultimo
          THEN BEGIN
            WRITELN('hay ',actual-1,' registros activos',
                    'en un fichero que los prevee',
                    ultimo-1,'!');
            WRITELN('no puedo proseguir.');
            WRITELN('pulse <RETURN>');
            READLN;
          EXIT(program);
         END;  {de la with viejo^}
     UNTIL j>=1;
     SEEK(cliente,1);
     GET(cliente);
     cliente^.primero:=actual;
     SEEK(cliente,1);
     PUT(cliente);
     CLOSE(cliente);
     CLOSE(indice);
     CLOSE(viejo);
END.
```

AL_PREPARA

```
PROGRAM al_prepara;
TYPE
     cli=RECORD
         CASE BOOLEAN OF
           TRUE:(cliente  :STRING[31];
```

```
              direccion:STRING[33];
              ciudad   :STRING[15];
              telefono :INTEGER[8];
              prefijo  :STRING[4];
              tipo     :CHAR;
              izquierdo:INTEGER;
              derecho  :INTEGER);
       FALSE:(primero  :INTEGER;
              maximo   :INTEGER);
   END;
VAR
   cliente:FILE OF cli;
       x,y,z:INTEGER;
       actual,ultimo:INTEGER;

  BEGIN
   REWRITE(cliente,'cliente.dat');
   WRITE('¿de cuantos registros estara compuesto',
        'el fichero?');
   READLN(ultimo);

   WITH cliente^
   DO BEGIN
    primero:=2;
    maximo:=ultimo+1;
    PUT(cliente);
    PUT(cliente);
    cliente:='';
    direccion:='';
    ciudad:='';
    telefono:=0;
    prefijo:='0000';
    tipo:=' ';
    FOR X:=1 TO ultimo DO PUT(cliente);
 END;
 CLOSE(cliente,LOCK);
END.
```


AL_GESTION

```
PROGRAM al_gestion;
TYPE
    cli=RECORD
       CASE BOOLEAN OF
         TRUE:(cliente  :STRING[31];
               direccion:STRING[33];
               ciudad   :STRING[15];
               telefono :INTEGER[8];
               prefijo  :STRING[4];
               tipo     :CHAR;
               izquierdo:INTEGER;
               derecho  :INTEGER);
        FALSE:(primero  :INTEGER;
               maximo   :INTEGER);
    END;
VAR
    cliente:FILE OF cli;
      hecho,enc,fuera,fin:BOOLEAN;
      partida,termino,alfa,nominativo:STRING[33];
      car,sw,que_quiere:CHAR;
      x,y,z,j:INTEGER;
      st:STRING[1];
      grande:INTEGER[15];
      p,m,x_pos,y_pos,esto,inicio,alt,pequeno,
      proximo,actual,ultimo:INTEGER;

PROCEDURE borrado_pantalla;
BEGIN
  WRITE(CHR$(28));
END;

PROCEDURE teclado;
BEGIN
 CASE sw OF
        '1':pequeno:=0;
        '2':grande:=0;
        '3':alfa:='';
  END;
  GOTOXY(x_pos,y_pos);
```

```
REPEAT
 READ(KEYBOARD,car);
 IF NOT EOLN(KEYBOARD)
 THEN BEGIN
   IF ORD(car)=8
    THEN BEGIN
      GOTOXY(x_pos,y_pos);
      CASE sw OF
       '1':BEGIN
               PEQUENO:=PEQUENO div 10;
               WRITE(PEQUENO,' ');
               GOTOXY(x_pos,y_pos);
                WRITE(pequeno);
            END;
        '2':BEGIN
               grande:=grande DIV 10;
          WRITE(grande,' ');
          GOTOXY(x_pos,y_pos);
          WRITE(grande);
         END;
   '3':BEGIN
              DELETE(alfa,LEGTH(alfa),1);
              WRITE(alfa,' ');
              GOTOXY(x_pos,y_pos);
              WRITE(alfa);
        END;
    END;
 END
 ELSE BEGIN
  IF sw IN['1','2']
  THEN BEGIN
    IF car
    THEN pequeno:=pequeno*10+ORD(car)-48
    ELSE grande:=grande*10+ORD(car)-48;
    WRITE(car);
  END
  ELSE WRITE(CHR$(7));
 END  {del campo numerico}
 ELSE BEGIN
  IF car IN[' '..'"']
  THEN BEGIN
```



```
      st:=' ';
      st[1]:=car;
      alfa:=CONCAT(alfa,st);
      WRITE(car);
     END
     ELSE WRITE(CHR$(7));
   END; {del campo alfabetico}
  END;  {si no tecla de retorno}
 END;   {no es return}
UNTIL EOLN(KEYBOARD);
END;

PROCEDURE menu;
BEGIN
 borrado_pantalla;
    WRITELN;
    WRITELN('1 --> carga de datos');
    WRITELN;
    WRITELN('2 --> lectura con el numero de registro');
    WRITELN;
    WRITELN('3 --> lectura con el nombre de cliente');
    WRITELN;
    WRITELN('4 --> modificacion de un registro');
    WRITELN;
    WRITELN('5 --> borrado de un registro');
    WRITELN;
    WRITELN('9 --> fin del programa');
    WRITELN;
    READ(que_quiere);WRITELN;
   END;

PROCEDURE visualizacion;
BEGIN
 WITH cliente^
 DO BEGIN
      GOTOXY(13,4);
      WRITE(cliente^.cliente);
      IF tipo<>' ' THEN WRITE(' borrado')
                   ELSE WRITE('        ');
     GOTOXY(13,6);
     WRITE(direccion);
```

```
      GOTOXY(13,8);
      WRITE(ciudad);
      GOTOXY(13,10);
      WRITE(telefono);
      GOTOXY(13,12);
      WRITE(prefijo);
      GOTOXY(13,14);
      WRITE(izquierdo);
      GOTOXY(33,14);
      WRITE(derecho);
  END;
END;

PROCEDURE mascara;
BEGIN
      GOTOXY(0,4);
      WRITE('cliente      :');
      GOTOXY(0,6);
      WRITE('direccion    :');
      GOTOXY(0,8);
      WRITE('ciudad       :');
      GOTOXY(0,10);
      WRITE('telefono     :');
      GOTOXY(0,12);
      WRITE('prefijo      :');
      GOTOXY(0,14);
      WRITE('izquierdo    :');
      GOTOXY(20,14);
      WRITE('derecho      :');
END;

PROCEDURE clave;
BEGIN
  enc:=FALSE;
  fin:=FALSE;
  esto:=2;
  j:=2;
  WITH cliente^
  DO BEGIN
   REPEAT
    IF j=0
```



```
     THEN BEGIN
       fin:=TRUE;
       proximo:=esto;
     END;
     ELSE BEGIN
     SEEK(cliente,j);
     GET(cliente);
     esto:=j;
     IF nominativo<cliente
      THEN j:=izquierdo
      ELSE IF nominativo=cliente
           THEN BEGIN
               enc:=TRUE;
               proximo:=derecho;
             END
             ELSE IF derecho>0 THEN j:=derecho
                                ELSE j:=0;
           END;
       UNTIL enc OR fin;
    END;   {del with cliente^}
   IF NOT fin
    THEN BEGIN
      SEEK(cliente,j);
      GET(cliente);
    END;
 END;

PROCEDURE modificacion;
BEGIN
  borrado_pantalla;
  GOTOXY(0,20);
  WRITE('clave a buscar: ');
  sw:='3';
  x_pos:=23;
  y_pos:=20;
  teclado;
  nominativo:=alfa;
  clave;
  IF NOT fin
  THEN BEGIN
    mascara;
```

```
visualizacion;
GOTOXY(0,18);
WRITE('pulse <ESC> para modificar el campo,',
      '<RETURN> para continuar');

GOTOXY(13,6);
READ(respuesta);
IF respuesta=CHR$(27)
THEN BEGIN
END;

GOTOXY(13,8);
READ(respuesta);
IF respuesta=CHR$(27)
THEN BEGIN
 GOTOXY(13,6);
 WRITE('                              ');
 sw:='3';
 x_pos:=13;
 y_pos:=6;
 teclado;
 cliente^.direccion;
END;

  GOTOXY(13,8);
  READ(respuesta);
  IF respuesta=CHR$(27);
  THEN BEGIN
  GOTOXY(13,8);
  WRITE('                    ');
  sw:='3';
  x_pos:=13;
  y_pos:=8;
  teclado;
  cliente^.ciudad:=alfa;
END;

GOTOXY(13,10);
READ(respuesta);
IF respuesta=CHR$(27)
THEN BEGIN
```



```
       GOTOXY(13,10);
       WRITE('                        ');
       sw:='2';
       x_pos:=13;
       y_pos:=10;
       teclado;
       cliente^.telefono:=grande;
     END;

     GOTOXY(13,12);
     READ(respuesta);
     IF respuesta=CHR$(27)
     THEN BEGIN
       GOTOXY(13,12);
       WRITE('                        ');
       sw:='3';
       x_pos:=13;
       y_pos:=12;
       teclado;
       cliente^.prefijo:=alfa;
     END;
   SEEK(cliente,esto);
   PUT(cliente);
   READ(respuesta);
   IF respuesta=CHR$(27)
END

ELSE BEGIN
   GOTOXY(0,20);
   WRITE('no existe este registro,pulse <RETURN>');
   READLN;
  END;
END;

PROCEDURE borrado;
BEGIN
      borrado_pantalla;
      GOTOXY(0,20);
      WRITE('clave a buscar: ');
      sw:='3';
      x_pos:=23;
      y_pos:=20;
```

```
      teclado;
      nominativo:=alfa;
      clave;
      IF NOT fin
      THEN BEGIN
        mascara;
        visualizazion;
        GOTOXY(0,20);
        WRITE('¿quiere borrarlo?(S/N)');
        READ(respuesta);
        IF respuesta IN ['S','s']
        THEN BEGIN
          cliente^.tipo:='C'
          SEEK(cliente,esto);
          PUT(cliente);
        END;
    END
    ELSE BEGIN
     GOTOXY(0,20);
     WRITE('no existe este registro,pulse <RETURN>');
     READLN;
    END;
 END;

PROCEDURE busqueda;
BEGIN
  WITH cliente^
  DO BEGIN
      fuera:=FALSE;
      REPEAT
        esto:=ABS(proximo);
        SEEK(cliente,esto);
        GET(cliente);
        IF (izquierdo<>0) AND (proximo>=0)
        THEN proximo:=izquierdo;
        ELSE BEGIN
        fuera:=TRUE;
        proximo:=derecho;
        END;
      UNTIL fuera;
    END;{with cliente^}
```



```
PROCEDURE nombre;
BEGIN
  GOTOXY(0,20);
  WRITE('nombre del primer cliente  :');
  sw:='3';
  x_pos:=32;
  y_pos:=20;
  teclado;
  partida:=alfa;
  GOTOXY(0,20);
  WRITE('nombre del ultimo cliente  :',
       '                             ');
  sw:='3';
  x_pos:=32;
  y_pos:=20;
  teclado;
  termino:=alfa;
  nominativo:=partida;
  clave;
  IF fin
  THEN BEGIN
   busqueda;
   fin:=FALSE;
  END;
  REPEAT
   borrado_pantalla;
   mascara;
   visualizacion;
   GOTOXY(0,20);
   esto:=proximo;
   IF esto<>0 THEN busqueda;
   WRITE('<ESPACIO> para continuar,',
         '<ESC> para terminar');
   READ(respuesta);
  UNTIL(respuesta=CHR$(27))
     OR(esto=0)
     OR(cliente^.cliente>termino)
 END;

PROCEDURE act_nudo;
BEGIN
```

```
WITH cliente^
DO BEGIN
      hecho:=FALSE;
      IF actual=2
      THEN BEGIN   {si primer registro}
       izquierdo:=0;
       derecho:=0;
END
ELSE BEGIN
   p:=2;
   REPEAT
      SEEK(cliente,p);
      GET(cliente);
      IF nominativo>cliente
      THEN BEGIN
        IF derecho>0
          THEN p:=derecho
          ELSE BEGIN
              hecho:=TRUE;
              m:=derecho;
              derecho:=actual;
              SEEK(cliente,p);
              PUT(cliente);
              derecho:=m;
              izquierdo:=0;
           END;
       END
       ELSE BEGIN
       IF izquierdo<>0
         THEN p:=izquierdo;
         ELSE BEGIN
             hecho:=TRUE;
             izquierdo:=actual;
             SEEK(cliente,p);
             PUT(cliente);
             izquierdo:=0;
             derecho:=-p;
          END;
      END;
    UNTIL hecho;
  END;
```



```
        actual:=actual+1;
     END; {with cliente^}
  END;

  PROCEDURE lista;
  BEGIN
    fin:=FALSE;
    IF (inicio<actual)
       AND (inicio>0)
       THEN BEGIN
          SEEK(cliente,inicio);
          GET(cliente);
          inicio:=SUCC(inicio);
       END
       ELSE fin:=TRUE;
     END;

  PROCEDURE lectura;
   BEGIN
    GOTOXY(0,20);
    WRITE('numero del primer registro: ');
    sw:='1';
    x_pos:=32;
    y_pos:=20;
    teclado;
    inicio:=pequeno+1
    GOTOXY(0,20);
    WRITE('numero del ultimo registro: ');
    teclado;
    alt:=pequeno+1;
    IF alt>actual-1 THEN alt:=actual-1;
    lista;
    IF NOT fin
    THEN REPEAT
      borrado_pantalla;
      mascara;
      visualizacion;
      GOTOXY(0,20);
      lista;
      WRITE('<ESPACIO> para continuar,',
            '<ESC> para terminar');
```

```
    READ(respuesta);
   UNTIL  (respuesta=CHR$(27))
       OR  fin
       OR (inicio)alt+1);
END;

PROCEDURE carga;
BEGIN
  borrado_pantalla;
  WRITELN('la carga parte con el numero',
          actual-1);
  IF actual>2
  THEN BEGIN
        SEEK(cliente,PRED(actual));
        GET(cliente);
        WRITELN('el registro anterior escrito',
                 'es el siguiente:');
        mascara;
        visualizacion;
        WRITELN;
    END;
  WRITELN('pulse <RETURN> para iniciar la carga');
  READLN;
  REPEAT
    borrado_pantalla;
  GOTOXY(0,20);
  WRITELN('para terminar,responda "zzzz" a la',
         'peticion del nombre del cliente');
  mascara;
  sw:='3';
  x_pos:=13;
  y_pos:=4;
  teclado;
  nominativo:=alfa;
  IF nominativo<>'zzzz'
   THEN BEGIN
     IF actual>ultimo
     THEN BEGIN
          WRITE('no puedo seguir.se ha superado',
                'la dimension maxima del fichero.',
                'pulse <RETURN>');
```



```
      READLN;
      EXIT(carga);
    END;
   clave;
   IF NOT fin
    THEN BEGIN
     GOTOXY(0,20);
     WRITE('este cliente ya existe.no puedo',
           'insertarlo.pulse <RETURN>');
     READLN;
     EXIT(carga);
    END;
  WITH cliente^
  DO BEGIN
    act_nudo;
      cliente:=nominativo;
      sw:='3';
      x_pos:=13;
      y_pos:=6;
      teclado;
      direccion:=alfa;
      sw:='3';
      y_pos:=8;
      teclado;
      ciudad:=alfa;
      sw:='2';
      y_pos:=10;
      teclado;
      telefono:=grande;
      sw:='3';
      x_pos:=13;
      y_pos:=12;
      teclado;
      prefijo:=alfa;
      tipo:=' ';
      SEEK(cliente,actual-1);
      PUT(cliente);
      primero:=actual;
      maximo:=ultimo;
      SEEK(cliente,1);
      PUT(cliente);
```

```
        END;  {de la with cliente}
      END;     {si cliente}
    UNTIL nominativo='zzzz';
   END;

  BEGIN
  RESET(cliente,'cliente.dat');
  SEEK(cliente,1);
  GET(cliente);
  actual:=cliente^.primero;
  ultimo:=cliente^.maximo;
  REPEAT
    menu;
    CASE que_quiere OF
         '1':carga;
         '2':lectura;
         '3':nombre;
         '4':modificacion;
         '5':borrado;
       END;
    UNTIL que_quiere='9';
    CLOSE(cliente.LOCK);
 END.
```

AL_COMPACTA

```
PROGRAM al_compacta;
TYPE
     cli=RECORD
        CASE BOOLEAN OF
          TRUE:(cliente   :STRING[31];
                direccion:STRING[33];
                ciudad    :STRING[15];
                telefono  :INTEGER[8];
                prefijo   :STRING[4];
                tipo      :CHAR;
                izquierdo:INTEGER;
                derecho   :INTEGER);
         FALSE:(primero   :INTEGER;
                maximo    :INTEGER);
    END;

VAR viejo,cliente:FILE OF cli;
    nominativo:STRING[31];
    hecho:BOOLEAN;
    m,p,x,y,z,actual,ultimo:INTEGER;
```



```
PROCEDURE act_nudo;
BEGIN
  WITH cliente^
  DO BEGIN
     hecho:=FALSE;
     IF actual=2
     THEN BEGIN  {si primer registro}
        izquierdo:=0;
        derecho:=0;
     END
     ELSE BEGIN
      p:=2;
      REPEAT
        SEEK(cliente,p);
        GET(cliente);
        IF nominativo>cliente
      THEN BEGIN
        IF derecho>0
        THEN p:=derecho;
        ELSE BEGIN
         hecho:=TRUE;
         m:=derecho;
         derecho:=actual;
         SEEK(cliente,p);
         PUT(cliente);
         derecho:=m;
         izquierdo:=0;
        END;
       END;
     ELSE BEGIN
          IF IZQUIERDO<>0
               THEN p:=izquierdo
               ELSE BEGIN
               hecho:=TRUE;
               izquierdo:=actual;
                 SEEK(cliente,p);
                 PUT(cliente);
                 izquierdo:=0;
                 derecho:=-p;
              END;
          END;
        UNTIL hecho;
      END;
      actual:=actual+1;
    END;   {WITH indice^}
END;

BEGIN
 RESET(viejo,'cliente.vie');
 RESET(cliente,'cliente.dat');
 SEEK(cliente,1);
 GET(cliente);
```

```
SEEK(viejo,1);
GET(viejo);
x:=cliente^.maximo;
y:=viejo^.primero;
GET(viejo);
actual:=2;
z:=2;
REPEAT
 WITH viejo^
 DO BEGIN
       IF tipo='C'
       THEN WRITELN(cliente:15,' borrado')
       ELSE BEGIN
       IF actual-1 > x
       THEN BEGIN
         WRITELN('hay ',actual-1,' registros',
                 'activos enun fichero que los',
                 'prevee ',x,'!');
         WRITELN('no puedo proseguir');
         WRITELN('pulse <RETURN>');
         READLN;
         EXIT(program);
       END;
       ELSE BEGIN
        nominativo:=cliente;
        act(nudo;
        cliente^.cliente
        cliente^.dirreccion:=direccion;
        cliente^.ciudad:=ciudad;
        cliente^.telefono:=telefono;
        cliente^.prefijo:=prefijo;
        cliente^.tipo:=' ';
        WRITELN(cliente:15);
        SEEK(cliente,actual-1);
             PUT(cliente);
             actual=actual+1
           END;
        END;
        z:=z+1:
        GET(viejo);
    END;  {de la with viejo^}
 UNTIL z>=y;

SEEK(cliente,1);
GET(cliente);
cliente^.primero:=actual;
SEEK(cliente,1);
PUT(cliente);
CLOSE(cliente,LOCK);
CLOSE(viejo);
END.
```



La complejidad del tratamiento (adquisición, actualización y acceso) de datos es más que considerable, lo cual implica también que los programas que se encargan de hacer posibles todas estas operaciones son igualmente complejos.

En este segundo volumen de la Biblioteca Básica Informática dedicado a los bancos de datos vamos a finalizar el estudio de los programas necesarios para gestionar un banco de datos, desde BASIC o Pascal, comenzado en el volumen anterior. Esto nos servirá, a la vez, como preparación para el acercamiento a uno de los programas comerciales de gestión de datos más populares y conocidos: el dBASE (II y III), que trataremos en un próximo volumen de la B.B.I.

395 pts.
(incluido IVA)